Heroku CLI Commands

Last updated 30 July 2019

These are the help texts for each of the core Heroku CLI commands. You can also see this text in your terminal with `heroku help`, `heroku --help`, or `heroku -h`.

heroku access

list who has access to an app

```
USAGE
  $ heroku access

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --json               output in json format
```

heroku access:add EMAIL

add new users to your app

```
USAGE
  $ heroku access:add EMAIL

OPTIONS
  -a, --app=app                  (required) app to run command against
  -p, --permissions=permissions  list of permissions comma separated
  -r, --remote=remote            git remote of app to use

EXAMPLES
  $ heroku access:add user@email.com --app APP # add a collaborator to your app
  $ heroku access:add user@email.com --app APP --permissions
  deploy,manage,operate # permissions must be comma separated
```

heroku access:remove EMAIL

remove users from a team app

```
USAGE
  $ heroku access:remove EMAIL

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLES
  $ heroku access:remove user@email.com --app APP
```

heroku access:update EMAIL

update existing collaborators on an team app

```
USAGE
  $ heroku access:update EMAIL

OPTIONS
  -a, --app=app                  (required) app to run command against

  -p, --permissions=permissions  comma-delimited list of permissions to update
                                 (deploy,manage,operate)

  -r, --remote=remote            git remote of app to use

EXAMPLES
  $ heroku access:update user@email.com --app APP --permissions
  deploy,manage,operate
```

heroku addons [--all|--app APP]

lists your add-ons and attachments

```
USAGE
  $ heroku addons [--all|--app APP]

OPTIONS
  -A, --all            show add-ons and attachments for all accessible apps
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
  --json               return add-ons in json format

DESCRIPTION
  The default filter applied depends on whether you are in a Heroku app
  directory. If so, the --app flag is implied. If not, the default of --all
  is implied. Explicitly providing either flag overrides the default
  behavior.

EXAMPLES
  $ heroku addons --all
  $ heroku addons --app acme-inc-www
```

heroku addons:attach ADDON_NAME

attach an existing add-on resource to an app

```
USAGE
  $ heroku addons:attach ADDON_NAME

OPTIONS
  -a, --app=app            (required) app to run command against
  -r, --remote=remote      git remote of app to use
  --as=as                  name for add-on attachment
  --confirm=confirm        overwrite existing add-on attachment with same name
  --credential=credential  credential name for scoped access to Heroku Postgres
```

heroku addons:create SERVICE:PLAN

create a new add-on resource

```
USAGE
  $ heroku addons:create SERVICE:PLAN

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --as=as              name for the initial add-on attachment

  --confirm=confirm    overwrite existing config vars or existing add-on
                       attachments

  --name=name          name for the add-on resource

  --wait               watch add-on creation status and exit when complete
```

heroku addons:destroy [ADDON]... [flags]

permanently destroy an add-on resource

```
USAGE
  $ heroku addons:destroy [ADDON]... [flags]

OPTIONS
  -a, --app=app          app to run command against
  -c, --confirm=confirm
  -f, --force            allow destruction even if connected to other apps
  -r, --remote=remote    git remote of app to use
```

heroku addons:detach ATTACHMENT_NAME

detach an existing add-on resource from an app

```
USAGE
  $ heroku addons:detach ATTACHMENT_NAME

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku addons:docs ADDON

open an add-on's Dev Center documentation in your browser

```
USAGE
  $ heroku addons:docs ADDON

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
  --show-url           show URL, do not open browser
```

heroku addons:downgrade ADDON [PLAN]

change add-on plan

```
USAGE
  $ heroku addons:downgrade ADDON [PLAN]

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  See available plans with `heroku addons:plans SERVICE`.

  Note that `heroku addons:upgrade` and `heroku addons:downgrade` are the same.
  Either one can be used to change an add-on plan up or down.

  https://devcenter.heroku.com/articles/managing-add-ons

EXAMPLE
  Upgrade an add-on by service name:
  $ heroku addons:upgrade heroku-redis:premium-2

  Upgrade a specific add-on:
  $ heroku addons:upgrade swimming-briskly-123 heroku-redis:premium-2
```

heroku addons:info ADDON

show detailed add-on resource and attachment information

```
USAGE
  $ heroku addons:info ADDON

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku addons:open ADDON

open an add-on's dashboard in your browser

```
USAGE
  $ heroku addons:open ADDON

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
  --show-url           show URL, do not open browser
```

heroku addons:plans SERVICE

list all available plans for an add-on services

```
USAGE
  $ heroku addons:plans SERVICE

OPTIONS
  --json  output in json format
```

heroku addons:rename ADDON NEW_NAME

rename an add-on

```
USAGE
  $ heroku addons:rename ADDON NEW_NAME

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku addons:services

list all available add-on services

```
USAGE
  $ heroku addons:services

OPTIONS
  --json  output in json format
```

heroku addons:upgrade ADDON [PLAN]

change add-on plan

```
USAGE
  $ heroku addons:upgrade ADDON [PLAN]

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  See available plans with `heroku addons:plans SERVICE`.

  Note that `heroku addons:upgrade` and `heroku addons:downgrade` are the same.
  Either one can be used to change an add-on plan up or down.

  https://devcenter.heroku.com/articles/managing-add-ons

EXAMPLE
  Upgrade an add-on by service name:
  $ heroku addons:upgrade heroku-redis:premium-2

  Upgrade a specific add-on:
  $ heroku addons:upgrade swimming-briskly-123 heroku-redis:premium-2
```

heroku addons:wait ADDON

show provisioning status of the add-ons on the app

```
USAGE
  $ heroku addons:wait ADDON

OPTIONS
  -a, --app=app                  app to run command against
  -r, --remote=remote            git remote of app to use
  --wait-interval=wait-interval  how frequently to poll in seconds
```

heroku apps

list your apps

```
USAGE
  $ heroku apps

OPTIONS
  -A, --all          include apps in all teams
  -p, --personal     list apps in personal account when a default team is set
  -s, --space=space  filter by space
  -t, --team=team    team to use
  --json             output in json format

EXAMPLES
  $ heroku apps
  === My Apps
  example
  example2

  === Collaborated Apps
  theirapp   other@owner.name
```

heroku apps:create [APP]

creates a new app

```
USAGE
  $ heroku apps:create [APP]

ARGUMENTS
  APP  name of app to create

OPTIONS
  -b, --buildpack=buildpack  buildpack url to use for this app
  -n, --no-remote            do not create a git remote
  -r, --remote=remote        the git remote to create, default "heroku"
  -s, --stack=stack          the stack to create the app on
  -t, --team=team            team to use
  --addons=addons            comma-delimited list of addons to install
  --json                     output in json format
  --region=region            specify region for the app to run in
  --space=space              the private space to create the app in
  --ssh-git                  use SSH git protocol for local git remote

EXAMPLES
  $ heroku apps:create
  Creating app... done, stack is cedar-14
  https://floating-dragon-42.heroku.com/ |
  https://git.heroku.com/floating-dragon-42.git

  # or just
  $ heroku create

  # use a heroku.yml manifest file
  $ heroku apps:create --manifest

  # specify a buildpack
  $ heroku apps:create --buildpack https://github.com/some/buildpack.git

  # specify a name
  $ heroku apps:create example

  # create a staging app
  $ heroku apps:create example-staging --remote staging

  # create an app in the eu region
  $ heroku apps:create --region eu
```

heroku apps:destroy

permanently destroy an app

```
USAGE
  $ heroku apps:destroy

OPTIONS
  -a, --app=app          app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use

DESCRIPTION
  This will also destroy all add-ons on the app.
```

heroku apps:errors

view app errors

```
USAGE
  $ heroku apps:errors

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --dyno               show only dyno errors
  --hours=hours        number of hours to look back (default 24)
  --json               output in json format
  --router             show only router errors
```

heroku apps:favorites

list favorited apps

```
USAGE
  $ heroku apps:favorites

OPTIONS
  --json  output in json format
```

heroku apps:favorites:add

favorites an app

```
USAGE
  $ heroku apps:favorites:add

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku apps:favorites:remove

unfavorites an app

```
USAGE
  $ heroku apps:favorites:remove

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku apps:info

show detailed app information

```
USAGE
  $ heroku apps:info

OPTIONS
  -a, --app=app        app to run command against
  -j, --json
  -r, --remote=remote  git remote of app to use
  -s, --shell          output more shell friendly key/value pairs

DESCRIPTION
  $ heroku apps:info
  === example
  Git URL:   https://git.heroku.com/example.git
  Repo Size: 5M
  ...

  $ heroku apps:info --shell
  git_url=https://git.heroku.com/example.git
  repo_size=5000000
  ...
```

heroku apps:join

add yourself to a team app

```
USAGE
  $ heroku apps:join

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku apps:leave

remove yourself from a team app

```
USAGE
  $ heroku apps:leave

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku apps:lock

prevent team members from joining an app

```
USAGE
  $ heroku apps:lock

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku apps:open [PATH]

open the app in a web browser

```
USAGE
  $ heroku apps:open [PATH]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLES
  $ heroku open -a myapp
  # opens https://myapp.herokuapp.com

  $ heroku open -a myapp /foo
  # opens https://myapp.herokuapp.com/foo
```

heroku apps:rename NEWNAME

rename an app

```
USAGE
  $ heroku apps:rename NEWNAME

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --ssh-git            use ssh git protocol instead of https

DESCRIPTION
  This will locally update the git remote if it is set to the old app.

EXAMPLES
  $ heroku apps:rename --app oldname newname
  https://newname.herokuapp.com/ | https://git.heroku.com/newname.git
  Git remote heroku updated
```

heroku apps:stacks

show the list of available stacks

```
USAGE
  $ heroku apps:stacks

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku apps:stacks:set STACK

set the stack of an app

```
USAGE
  $ heroku apps:stacks:set STACK

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLES
  $ heroku stack:set cedar-14 -a myapp
  Stack set. Next release on myapp will use cedar-14.
  Run git push heroku master to create a new release on myapp.
```

heroku apps:transfer RECIPIENT

transfer applications to another user or team

```
USAGE
  $ heroku apps:transfer RECIPIENT

ARGUMENTS
  RECIPIENT  user or team to transfer applications to

OPTIONS
  -a, --app=app        app to run command against
  -l, --locked         lock the app upon transfer
  -r, --remote=remote  git remote of app to use
  --bulk               transfer applications in bulk

EXAMPLES
  $ heroku apps:transfer collaborator@example.com
  Transferring example to collaborator@example.com... done

  $ heroku apps:transfer acme-widgets
  Transferring example to acme-widgets... done

  $ heroku apps:transfer --bulk acme-widgets
  ...
```

heroku apps:unlock

unlock an app so any team member can join

```
USAGE
  $ heroku apps:unlock

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku auth:2fa

check 2fa status

```
USAGE
  $ heroku auth:2fa

ALIASES
  $ heroku 2fa
  $ heroku twofactor
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/auth/2fa/index.ts)

heroku auth:2fa:disable

disables 2fa on account

```
USAGE
  $ heroku auth:2fa:disable

ALIASES
  $ heroku twofactor:disable
  $ heroku 2fa:disable

EXAMPLES
  $ heroku auth:2fa:disable
  Disabling 2fa on me@example.com... done
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/auth/2fa/disable.ts)

heroku auth:2fa:generate-recovery-codes

generates 2fa recovery codes

```
USAGE
  $ heroku auth:2fa:generate-recovery-codes

DESCRIPTION
  If you lose access to your two-factor device, e.g. you lose your phone or it
  is wiped, you can still log in to your account. When prompted for the second
  factor after entering your account password, choose "Enter a Recovery Code."
  You can then enter one of your recovery codes instead of a token from your
  two-factor device. Note that each recovery code can only be used once.

  Running this command will replace existing codes.

ALIASES
  $ heroku twofactor:generate-recovery-codes
  $ heroku 2fa:generate-recovery-codes
  $ heroku auth:2fa:generate

EXAMPLES
  $ heroku auth:2fa:generate
  Password: ********************
  Recovery codes:
  02799c92ab3ba7c7
  09aea052a72b6a22
  361e00bb82c7cbd4
  588ac05dec23952c
  6020ef9ec364066b
  6cfd923315875e78
  7c576b935eafc452
  8c00eeb258ee565e
  a37c5c6985f56e66
  f82e7c2a50737494
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/auth/2fa/generate-recovery-codes.ts)

heroku auth:login

login with your Heroku credentials

```
USAGE
  $ heroku auth:login

OPTIONS
  -e, --expires-in=expires-in  duration of token in seconds (default 1 year)
  -i, --interactive            login with username/password

  --browser=browser            browser to open SSO with (example: "firefox",
                               "safari")

ALIASES
  $ heroku login
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/auth/login.ts)

heroku auth:logout

clears local login credentials and invalidates API session

```
USAGE
  $ heroku auth:logout

ALIASES
  $ heroku logout
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/auth/logout.ts)

heroku auth:token

outputs current CLI authentication token.

```
USAGE
  $ heroku auth:token

OPTIONS
  -h, --help  show CLI help

DESCRIPTION
  By default, the CLI auth token is only valid for 1 year. To generate a
  long-lived token, use heroku authorizations:create
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/auth/token.ts)

heroku auth:whoami

display the current logged in user

```
USAGE
  $ heroku auth:whoami

ALIASES
  $ heroku whoami
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/auth/whoami.ts)

heroku authorizations

list OAuth authorizations

```
USAGE
  $ heroku authorizations

OPTIONS
  -j, --json  output in json format
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/authorizations/index.js)

heroku authorizations:create

create a new OAuth authorization

```
USAGE
  $ heroku authorizations:create

OPTIONS
  -S, --short                    only output token
  -d, --description=description  set a custom authorization description

  -e, --expires-in=expires-in    set expiration in seconds (default no
                                 expiration)

  -j, --json                     output in json format

  -s, --scope=scope              set custom OAuth scopes

DESCRIPTION
  This creates an authorization with access to your Heroku account.
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/authorizations/create.js)

heroku authorizations:info ID

show an existing OAuth authorization

```
USAGE
  $ heroku authorizations:info ID

OPTIONS
  -j, --json  output in json format
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/authorizations/info.js)

heroku authorizations:revoke ID

revoke OAuth authorization

```
USAGE
  $ heroku authorizations:revoke ID

ALIASES
  $ heroku authorizations:destroy
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/authorizations/revoke.js)

heroku authorizations:rotate ID

updates an OAuth authorization token

```
USAGE
  $ heroku authorizations:rotate ID
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/authorizations/rotate.js)

heroku authorizations:update ID

updates an OAuth authorization

```
USAGE
  $ heroku authorizations:update ID

OPTIONS
  -d, --description=description  set a custom authorization description
  --client-id=client-id          identifier of OAuth client to set
  --client-secret=client-secret  secret of OAuth client to set
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/authorizations/update.js)

heroku autocomplete [SHELL]

display autocomplete installation instructions

```
USAGE
  $ heroku autocomplete [SHELL]

ARGUMENTS
  SHELL  shell type

OPTIONS
  -r, --refresh-cache  refresh cache only (ignores displaying instructions)

EXAMPLES
  $ heroku autocomplete
  $ heroku autocomplete bash
  $ heroku autocomplete zsh
  $ heroku autocomplete --refresh-cache
```

See code: @heroku-cli/plugin-autocomplete (https://github.com/heroku/cli/blob/v7.27.0/packages/autocomplete/src/commands/autocomplete/index.ts)

heroku buildpacks

display the buildpacks for an app

```
USAGE
  $ heroku buildpacks

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/index.ts)

heroku buildpacks:add BUILDPACK

add new app buildpack, inserting into list of buildpacks if necessary

```
USAGE
  $ heroku buildpacks:add BUILDPACK

ARGUMENTS
  BUILDPACK  namespace/name of the buildpack

OPTIONS
  -a, --app=app        (required) app to run command against
  -i, --index=index    the 1-based index of the URL in the list of URLs
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/add.ts)

heroku buildpacks:clear

clear all buildpacks set on the app

```
USAGE
  $ heroku buildpacks:clear

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/clear.ts)

heroku buildpacks:info BUILDPACK

fetch info about a buildpack

```
USAGE
  $ heroku buildpacks:info BUILDPACK

ARGUMENTS
  BUILDPACK  namespace/name of the buildpack
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/info.ts)

heroku buildpacks:remove [BUILDPACK]

remove a buildpack set on the app

```
USAGE
  $ heroku buildpacks:remove [BUILDPACK]

ARGUMENTS
  BUILDPACK  namespace/name of the buildpack

OPTIONS
  -a, --app=app        (required) app to run command against

  -i, --index=index    the 1-based index of the URL to remove from the list of
                       URLs

  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/remove.ts)

heroku buildpacks:search [TERM]

search for buildpacks

```
USAGE
  $ heroku buildpacks:search [TERM]

ARGUMENTS
  TERM  search term that searches across name, namespace, and description

OPTIONS
  --description=description  buildpack description to filter on

  --name=name                buildpack names to filter on using a comma
                             separated list

  --namespace=namespace      buildpack namespaces to filter on using a comma
                             separated list
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/search.ts)

heroku buildpacks:set BUILDPACK

```
USAGE
  $ heroku buildpacks:set BUILDPACK

ARGUMENTS
  BUILDPACK  namespace/name of the buildpack

OPTIONS
  -a, --app=app        (required) app to run command against
  -i, --index=index    the 1-based index of the URL in the list of URLs
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/set.ts)

heroku buildpacks:versions BUILDPACK

list versions of a buildpack

```
USAGE
  $ heroku buildpacks:versions BUILDPACK

ARGUMENTS
  BUILDPACK  namespace/name of the buildpack
```

See code: @heroku-cli/plugin-buildpacks (https://github.com/heroku/cli/blob/v7.27.0/packages/buildpacks/src/commands/buildpacks/versions.ts)

heroku certs

list SSL certificates for an app

```
USAGE
  $ heroku certs

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/index.js)

heroku certs:add CRT KEY

add an SSL certificate to an app

```
USAGE
  $ heroku certs:add CRT KEY

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --bypass             bypass the trust chain completion step
  --domains=domains    domains to create after certificate upload
  --type=type          type to create, either 'sni' or 'endpoint'

DESCRIPTION
  Note: certificates with PEM encoding are also valid

EXAMPLES
  $ heroku certs:add example.com.crt example.com.key

  Certificate Intermediary:
  $ heroku certs:add intermediary.crt example.com.crt example.com.key
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/add.js)

heroku certs:auto

show ACM status for an app

```
USAGE
  $ heroku certs:auto

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/auto/index.js)

heroku certs:auto:disable

disable ACM for an app

```
USAGE
  $ heroku certs:auto:disable

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/auto/disable.js)

heroku certs:auto:enable

enable ACM status for an app

```
USAGE
  $ heroku certs:auto:enable

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/auto/enable.js)

heroku certs:auto:refresh

refresh ACM for an app

```
USAGE
  $ heroku certs:auto:refresh

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/auto/refresh.js)

heroku certs:chain

print an ordered & complete chain for a certificate

```
USAGE
  $ heroku certs:chain

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/chain.js)

heroku certs:generate DOMAIN

generate a key and a CSR or self-signed certificate

```
USAGE
  $ heroku certs:generate DOMAIN

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --area=area          sub-country area (state, province, etc.) of owner
  --city=city          city of owner
  --country=country    country of owner, as a two-letter ISO country code
  --keysize=keysize    RSA key size in bits (default: 2048)
  --now                do not prompt for any owner information
  --owner=owner        name of organization certificate belongs to
  --selfsigned         generate a self-signed certificate instead of a CSR
  --subject=subject    specify entire certificate subject

DESCRIPTION
  Generate a key and certificate signing request (or self-signed certificate)
  for an app. Prompts for information to put in the certificate unless --now
  is used, or at least one of the --subject, --owner, --country, --area, or
  --city options is specified.

EXAMPLES
  $ heroku certs:generate example.com
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/generate.js)

heroku certs:info

show certificate information for an SSL certificate

```
USAGE
  $ heroku certs:info

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --endpoint=endpoint  endpoint to check info on
  --name=name          name to check info on
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/info.js)

heroku certs:key

print the correct key for the given certificate

```
USAGE
  $ heroku certs:key

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  You must pass one single certificate, and one or more keys.
  The first key that signs the certificate will be printed back.

EXAMPLES
  $ heroku certs:key example.com.crt example.com.key
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/key.js)

heroku certs:remove

remove an SSL certificate from an app

```
USAGE
  $ heroku certs:remove

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --endpoint=endpoint  endpoint to remove
  --name=name          name to remove
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/remove.js)

heroku certs:rollback

rollback an SSL certificate from an app

```
USAGE
  $ heroku certs:rollback

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --endpoint=endpoint  endpoint to rollback
  --name=name          name to rollback
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/rollback.js)

heroku certs:update CRT KEY

update an SSL certificate on an app

```
USAGE
  $ heroku certs:update CRT KEY

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --bypass             bypass the trust chain completion step
  --endpoint=endpoint  endpoint to update
  --name=name          name to update

DESCRIPTION
  Note: certificates with PEM encoding are also valid

EXAMPLES
  $ heroku certs:update example.com.crt example.com.key

  Certificate Intermediary:
  $ heroku certs:update intermediary.crt example.com.crt example.com.key
```

See code: @heroku-cli/plugin-certs-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/certs-v5/commands/certs/update.js)

heroku ci

display the most recent CI runs for the given pipeline

```
USAGE
  $ heroku ci

OPTIONS
  -a, --app=app            app name
  -p, --pipeline=pipeline  name of pipeline
  --json                   output in json format
  --watch                  keep running and watch for new and update tests

EXAMPLE
  $ heroku ci --app murmuring-headland-14719
```

See code: @heroku-cli/plugin-ci (https://github.com/heroku/cli/blob/v7.24.3/packages/ci/src/commands/ci/index.ts)

heroku ci:config

display CI config vars

```
USAGE
  $ heroku ci:config

OPTIONS
  -a, --app=app            app to run command against
  -p, --pipeline=pipeline  pipeline
  -r, --remote=remote      git remote of app to use
  -s, --shell              output config vars in shell format
  --json                   output config vars in json format

DESCRIPTION
  Example:

       $ heroku ci:config --app murmuring-headland-14719 --json
```

heroku ci:config:get KEY

get a CI config var

```
USAGE
  $ heroku ci:config:get KEY

OPTIONS
  -a, --app=app            app to run command against
  -p, --pipeline=pipeline  pipeline
  -r, --remote=remote      git remote of app to use
  -s, --shell              output config var in shell format

DESCRIPTION
  Examples:

       $ heroku ci:config:get RAILS_ENV
       test
```

heroku ci:config:set

set CI config vars

```
USAGE
  $ heroku ci:config:set

OPTIONS
  -a, --app=app            app to run command against
  -p, --pipeline=pipeline  pipeline
  -r, --remote=remote      git remote of app to use

DESCRIPTION
  Examples:

       $ heroku ci:config:set RAILS_ENV=test
       Setting test config vars... done

       RAILS_ENV: test
```

heroku ci:config:unset

unset CI config vars

```
USAGE
  $ heroku ci:config:unset

OPTIONS
  -a, --app=app            app to run command against
  -p, --pipeline=pipeline  pipeline
  -r, --remote=remote      git remote of app to use

DESCRIPTION
  Examples:

       $ heroku ci:config:uset RAILS_ENV
       Unsetting RAILS_ENV... done
```

heroku ci:debug

opens an interactive test debugging session with the contents of the current directory

```
USAGE
  $ heroku ci:debug

OPTIONS
  -a, --app=app            app to run command against
  -p, --pipeline=pipeline  pipeline
  -r, --remote=remote      git remote of app to use
  --no-cache               start test run with an empty cache
  --no-setup               start test dyno without running test-setup

DESCRIPTION
  Example:

       $ heroku ci:debug
       Preparing source... done
       Creating test run... done
       Running setup and attaching to test dyno...

  ~ $
```

heroku ci:info TEST-RUN

show the status of a specific test run

```
USAGE
  $ heroku ci:info TEST-RUN

OPTIONS
  -a, --app=app            app name
  -p, --pipeline=pipeline  name of pipeline
  --node=node              the node number to show its setup and output

EXAMPLE
  $ heroku ci:info 1288 --app murmuring-headland-14719
```

See code: @heroku-cli/plugin-ci (https://github.com/heroku/cli/blob/v7.24.3/packages/ci/src/commands/ci/info.ts)

heroku ci:last

looks for the most recent run and returns the output of that run

```
USAGE
  $ heroku ci:last

OPTIONS
  -a, --app=app            app name
  -p, --pipeline=pipeline  name of pipeline
  --node=node              the node number to show its setup and output

EXAMPLE
  $ heroku ci:last --app murmuring-headland-14719 --node 100
```

See code: @heroku-cli/plugin-ci (https://github.com/heroku/cli/blob/v7.24.3/packages/ci/src/commands/ci/last.ts)

heroku ci:migrate-manifest

app-ci.json is deprecated. Run this command to migrate to app.json with an environments key.

```
USAGE
  $ heroku ci:migrate-manifest

DESCRIPTION
  Example:

        $ heroku ci:migrate-manifest
        Writing app.json file... done
        Deleting app-ci.json file... done
        Please check the contents of your app.json before committing to your repo
        You're all set! 🎉.
```

heroku ci:open

open the Dashboard version of Heroku CI

```
USAGE
  $ heroku ci:open

OPTIONS
  -a, --app=app            app to run command against
  -p, --pipeline=pipeline  pipeline
  -r, --remote=remote      git remote of app to use

DESCRIPTION
  opens a browser to view the Dashboard version of Heroku CI

       Example:

       $ heroku ci:open --app murmuring-headland-14719
```

heroku ci:rerun [NUMBER]

rerun tests against current directory

```
USAGE
  $ heroku ci:rerun [NUMBER]

OPTIONS
  -a, --app=app            app name
  -p, --pipeline=pipeline  name of pipeline

EXAMPLE
  $ heroku ci:rerun 985 --app murmuring-headland-14719
```

See code: @heroku-cli/plugin-ci (https://github.com/heroku/cli/blob/v7.24.3/packages/ci/src/commands/ci/rerun.ts)

heroku ci:run

run tests against current directory

```
USAGE
  $ heroku ci:run

OPTIONS
  -a, --app=app            app name
  -p, --pipeline=pipeline  name of pipeline

EXAMPLE
  $ heroku ci:run --app murmuring-headland-14719
```

See code: @heroku-cli/plugin-ci (https://github.com/heroku/cli/blob/v7.24.3/packages/ci/src/commands/ci/run.ts)

heroku clients

list your OAuth clients

```
USAGE
  $ heroku clients

OPTIONS
  -j, --json  output in json format
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/clients/index.js)

heroku clients:create NAME REDIRECT_URI

create a new OAuth client

```
USAGE
  $ heroku clients:create NAME REDIRECT_URI

OPTIONS
  -j, --json   output in json format
  -s, --shell  output in shell format
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/clients/create.js)

heroku clients:destroy ID

delete client by ID

```
USAGE
  $ heroku clients:destroy ID
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/clients/destroy.js)

heroku clients:info ID

show details of an oauth client

```
USAGE
  $ heroku clients:info ID

OPTIONS
  -j, --json   output in json format
  -s, --shell  output in shell format
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/clients/info.js)

heroku clients:rotate ID

rotate OAuth client secret

```
USAGE
  $ heroku clients:rotate ID

OPTIONS
  -j, --json   output in json format
  -s, --shell  output in shell format
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/clients/rotate.js)

heroku clients:update ID

update OAuth client

```
USAGE
  $ heroku clients:update ID

OPTIONS
  -n, --name=name  change the client name
  --url=url        change the client redirect URL
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/clients/update.js)

heroku commands

list all the commands

```
USAGE
  $ heroku commands

OPTIONS
  -h, --help  show CLI help
  -j, --json  output in json format
  --hidden    also show hidden commands
```

See code: @oclif/plugin-commands (https://github.com/oclif/plugin-commands/blob/v1.2.2/src/commands/commands.ts)

heroku config

display the config vars for an app

```
USAGE
  $ heroku config

OPTIONS
  -a, --app=app        (required) app to run command against
  -j, --json           output config vars in json format
  -r, --remote=remote  git remote of app to use
  -s, --shell          output config vars in shell format
```

See code: @heroku-cli/plugin-config (https://github.com/heroku/cli/blob/v7.24.0/packages/config/src/commands/config/index.ts)

heroku config:edit [KEY]

interactively edit config vars

```
USAGE
  $ heroku config:edit [KEY]

ARGUMENTS
  KEY  edit a single key

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  This command opens the app config in a text editor set by $VISUAL or $EDITOR.
  Any variables added/removed/changed will be updated on the app after saving
  and closing the file.

EXAMPLES
  # edit with vim
  $ EDITOR="vim" heroku config:edit
  # edit with emacs
  $ EDITOR="emacs" heroku config:edit
  # edit with pico
  $ EDITOR="pico" heroku config:edit
  # edit with atom editor
  $ VISUAL="atom --wait" heroku config:edit
```

See code: @heroku-cli/plugin-config (https://github.com/heroku/cli/blob/v7.24.0/packages/config/src/commands/config/edit.ts)

heroku config:get KEY...

display a single config value for an app

```
USAGE
  $ heroku config:get KEY...

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -s, --shell          output config vars in shell format

EXAMPLES
  $ heroku config:get RAILS_ENV
  production
```

See code: @heroku-cli/plugin-config (https://github.com/heroku/cli/blob/v7.24.0/packages/config/src/commands/config/get.ts)

heroku config:set

set one or more config vars

```
USAGE
  $ heroku config:set

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLES
  $ heroku config:set RAILS_ENV=staging
  Setting config vars and restarting example... done, v10
  RAILS_ENV: staging

  $ heroku config:set RAILS_ENV=staging RACK_ENV=staging
  Setting config vars and restarting example... done, v11
  RAILS_ENV: staging
  RACK_ENV:  staging
```

heroku config:unset

unset one or more config vars

```
USAGE
  $ heroku config:unset

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

ALIASES
  $ heroku config:remove

EXAMPLES
  $ heroku config:unset RAILS_ENV
  Unsetting RAILS_ENV and restarting example... done, v10
  $ heroku config:unset RAILS_ENV RACK_ENV
  Unsetting RAILS_ENV, RACK_ENV and restarting example... done, v10
```

See code: @heroku-cli/plugin-config (https://github.com/heroku/cli/blob/v7.24.0/packages/config/src/commands/config/unset.ts)

heroku container

Use containers to build and deploy Heroku apps

```
USAGE
  $ heroku container
```

heroku container:login

log in to Heroku Container Registry

```
USAGE
  $ heroku container:login

OPTIONS
  -v, --verbose

DESCRIPTION
  Usage:
           heroku container:login
```

heroku container:logout

log out from Heroku Container Registry

```
USAGE
  $ heroku container:logout

OPTIONS
  -v, --verbose
```

heroku container:pull

pulls an image from an app's process type

```
USAGE
  $ heroku container:pull

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -v, --verbose

DESCRIPTION
  Usage:
       heroku container:pull web        # Pulls the web image from the app
       heroku container:pull web worker # Pulls both the web and worker images
  from the app
       heroku container:pull web:latest # Pulls the latest tag from the web
  image
```

heroku container:push

builds, then pushes Docker images to deploy your Heroku app

```
USAGE
  $ heroku container:push

OPTIONS
  -R, --recursive              pushes Dockerfile.<process> found in current and
                               subdirectories

  -a, --app=app                (required) app to run command against

  -r, --remote=remote          git remote of app to use

  -v, --verbose

  --arg=arg                    set build-time variables

  --context-path=context-path  path to use as build context (defaults to
                               Dockerfile dir)

EXAMPLES
  heroku container:push web                          # Pushes Dockerfile to web
  process type
  heroku container:push worker                       # Pushes Dockerfile to
  worker process type
  heroku container:push web worker --recursive       # Pushes Dockerfile.web and
  Dockerfile.worker
  heroku container:push --recursive                  # Pushes Dockerfile.*
  heroku container:push web --arg ENV=live,HTTPS=on  # Build-time variables
  heroku container:push --recursive --context-path . # Pushes Dockerfile.* using
  current dir as build context
```

heroku container:release

Releases previously pushed Docker images to your Heroku app

```
USAGE
  $ heroku container:release

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -v, --verbose

DESCRIPTION
  Usage:
       heroku container:release web                      # Releases the
  previously pushed web process type
       heroku container:release web worker               # Releases the
  previously pushed web and worker process types
```

heroku container:rm

remove the process type from your app

```
USAGE
  $ heroku container:rm

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Usage:
       heroku container:rm web        # Destroys the web container
       heroku container:rm web worker # Destroys the web and worker containers
```

heroku container:run

builds, then runs the docker image locally

```
USAGE
  $ heroku container:run

OPTIONS
  -a, --app=app        (required) app to run command against
  -p, --port=port      port the app will run on
  -r, --remote=remote  git remote of app to use
  -v, --verbose

DESCRIPTION
  Usage:
       heroku container:run web bash # Runs bash on the local web docker
  container
       heroku container:run worker   # Runs the container CMD on the local
  worker container
```

heroku domains

list domains for an app

```
USAGE
  $ heroku domains

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --json               output in json format

EXAMPLES
  $ heroku domains
  === example Heroku Domain
  example.herokuapp.com

  === example Custom Domains
  Domain Name      DNS Record Type  DNS Target
  ___________      _________________  ___________
  www.example.com  CNAME            www.example.herokudns.com
```

heroku domains:add HOSTNAME

add domain to an app

```
USAGE
  $ heroku domains:add HOSTNAME

OPTIONS
  -a, --app=app        (required) app to run command against
  -j, --json           output in json format
  -r, --remote=remote  git remote of app to use
  --wait
```

heroku domains:clear

remove all domains from an app

```
USAGE
  $ heroku domains:clear

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku domains:remove HOSTNAME

remove domain from an app

```
USAGE
  $ heroku domains:remove HOSTNAME

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku domains:wait [HOSTNAME]

wait for domain to be active for an app

```
USAGE
  $ heroku domains:wait [HOSTNAME]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku drains

display the log drains of an app

```
USAGE
  $ heroku drains

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --json               output in json format
```

heroku drains:add URL

adds a log drain to an app

```
USAGE
  $ heroku drains:add URL

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku drains:remove [URL|TOKEN]

removes a log drain from an app

```
USAGE
  $ heroku drains:remove [URL|TOKEN]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku dyno:kill DYNO

stop app dyno

```
USAGE
  $ heroku dyno:kill DYNO

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  stop app dyno or dyno type

EXAMPLES
  $ heroku ps:stop run.1828
  Stopping run.1828 dyno... done

  $ heroku ps:stop run
  Stopping run dynos... done
```

heroku dyno:resize

manage dyno sizes

```
USAGE
  $ heroku dyno:resize

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Called with no arguments shows the current dyno size.

  Called with one argument sets the size.
  Where SIZE is one of free|hobby|standard-1x|standard-2x|performance

  Called with 1..n TYPE=SIZE arguments sets the quantity per type.
```

heroku dyno:restart [DYNO]

restart app dynos

```
USAGE
  $ heroku dyno:restart [DYNO]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  if DYNO is not specified, restarts all dynos on app

EXAMPLES
  $ heroku ps:restart web.1
  Restarting web.1 dyno... done

  $ heroku ps:restart web
  Restarting web dynos... done

  $ heroku ps:restart
  Restarting dynos... done
```

heroku dyno:scale

scale dyno quantity up or down

```
USAGE
  $ heroku dyno:scale

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Appending a size (eg. web=2:Standard-2X) allows simultaneous scaling and
  resizing.

  Omitting any arguments will display the app's current dyno formation, in a
  format suitable for passing back into ps:scale.

EXAMPLES
  $ heroku ps:scale web=3:Standard-2X worker+1
  Scaling dynos... done, now running web at 3:Standard-2X, worker at
  1:Standard-1X.

  $ heroku ps:scale
  web=3:Standard-2X worker=1:Standard-1X
```

heroku dyno:stop DYNO

stop app dyno

```
USAGE
  $ heroku dyno:stop DYNO

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  stop app dyno or dyno type

EXAMPLES
  $ heroku ps:stop run.1828
  Stopping run.1828 dyno... done

  $ heroku ps:stop run
  Stopping run dynos... done
```

heroku features

list available app features

```
USAGE
  $ heroku features

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --json               output in json format
```

heroku features:disable FEATURE

disables an app feature

```
USAGE
  $ heroku features:disable FEATURE

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku features:enable FEATURE

enables an app feature

```
USAGE
  $ heroku features:enable FEATURE

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku features:info FEATURE

display information about a feature

```
USAGE
  $ heroku features:info FEATURE

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --json               output in json format
```

heroku git:clone [DIRECTORY]

clones a heroku app to your local machine at DIRECTORY (defaults to app name)

```
USAGE
  $ heroku git:clone [DIRECTORY]

ARGUMENTS
  DIRECTORY  where to clone the app

OPTIONS
  -a, --app=app        (required) the Heroku app to use
  -r, --remote=remote  the git remote to create, default "heroku"
  --ssh-git            use SSH git protocol

EXAMPLES
  $ heroku git:clone -a example
  Cloning into 'example'...
  remote: Counting objects: 42, done.
  ...
```

See code: @heroku-cli/plugin-git (https://github.com/heroku/cli/blob/v7.24.0/packages/git/src/commands/git/clone.ts)

heroku git:remote

adds a git remote to an app repo

```
USAGE
  $ heroku git:remote

OPTIONS
  -a, --app=app        the Heroku app to use
  -r, --remote=remote  the git remote to create
  --ssh-git            use SSH git protocol

DESCRIPTION
  extra arguments will be passed to git remote add

EXAMPLES
  # set git remote heroku to https://git.heroku.com/example.git
       $ heroku git:remote -a example

       # set git remote heroku-staging to
  https://git.heroku.com/example-staging.git
       $ heroku git:remote --remote heroku-staging -a example
```

See code: @heroku-cli/plugin-git (https://github.com/heroku/cli/blob/v7.24.0/packages/git/src/commands/git/remote.ts)

heroku help [COMMAND]

display help for heroku

```
USAGE
  $ heroku help [COMMAND]

ARGUMENTS
  COMMAND  command to show help for

OPTIONS
  --all  see all commands in CLI
```

See code: @oclif/plugin-help (https://github.com/oclif/plugin-help/blob/v2.2.0/src/commands/help.ts)

heroku join

add yourself to a team app

```
USAGE
  $ heroku join

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku keys

display your SSH keys

```
USAGE
  $ heroku keys

OPTIONS
  -l, --long  display full SSH keys
  --json      output in json format
```

heroku keys:add [KEY]

add an SSH key for a user

```
USAGE
  $ heroku keys:add [KEY]

OPTIONS
  -y, --yes  automatically answer yes for all prompts

DESCRIPTION
  if no KEY is specified, will try to find ~/.ssh/id_rsa.pub

EXAMPLES
  $ heroku keys:add
  Could not find an existing public key.
  Would you like to generate one? [Yn] y
  Generating new SSH public key.
  Uploading SSH public key /.ssh/id_rsa.pub... done

  $ heroku keys:add /my/key.pub
  Uploading SSH public key /my/key.pub... done
```

heroku keys:clear

remove all SSH keys for current user

```
USAGE
  $ heroku keys:clear
```

heroku keys:remove KEY

remove an SSH key from the user

```
USAGE
  $ heroku keys:remove KEY

EXAMPLES
  $ heroku keys:remove email@example.com
  Removing email@example.com SSH key... done
```

heroku labs

list experimental features

```
USAGE
  $ heroku labs

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
  --json               display as json
```

heroku labs:disable [FEATURE]

disables an experimental feature

```
USAGE
  $ heroku labs:disable [FEATURE]

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
  --confirm=confirm
```

See code: @heroku-cli/plugin-auth (https://github.com/heroku/cli/blob/v7.24.0/packages/auth/src/commands/labs/disable.ts)

heroku labs:enable FEATURE

enables an experimental feature

```
USAGE
  $ heroku labs:enable FEATURE

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku labs:info FEATURE

show feature info

```
USAGE
  $ heroku labs:info FEATURE

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
  --json               display as json
```

heroku leave

remove yourself from a team app

```
USAGE
  $ heroku leave

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku local [PROCESSNAME]

run heroku app locally

```
USAGE
  $ heroku local [PROCESSNAME]

OPTIONS
  -e, --env=env            location of env file (defaults to .env)
  -f, --procfile=procfile  use a different Procfile
  -p, --port=port          port to listen on

DESCRIPTION
  Start the application specified by a Procfile (defaults to ./Procfile)

ALIASES
  $ heroku local:start

EXAMPLE
  $ heroku local
  $ heroku local web
  $ heroku local web=2
  $ heroku local web=1,worker=2
```

See code: @heroku-cli/plugin-local (https://github.com/heroku/cli/blob/v7.26.2/src/commands/local/index.ts)

heroku local:run

run a one-off command

```
USAGE
  $ heroku local:run

OPTIONS
  -e, --env=env
  -p, --port=port

EXAMPLE
  $ heroku local:run bin/migrate
```

See code: @heroku-cli/plugin-local (https://github.com/heroku/cli/blob/v7.26.2/src/commands/local/run.ts)

heroku local:version

display node-foreman version

```
USAGE
  $ heroku local:version
```

See code: @heroku-cli/plugin-local (https://github.com/heroku/cli/blob/v7.26.2/src/commands/local/version.ts)

heroku lock

prevent team members from joining an app

```
USAGE
  $ heroku lock

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku logs

display recent log output

```
USAGE
  $ heroku logs

OPTIONS
  -a, --app=app        (required) app to run command against

  -d, --dyno=dyno      only show output from this dyno type (such as "web" or
                       "worker")

  -n, --num=num        number of lines to display

  -r, --remote=remote  git remote of app to use

  -s, --source=source  only show output from this source (such as "app" or
                       "heroku")

  -t, --tail           continually stream logs

  --force-colors       force use of colors (even on non-tty output)

DESCRIPTION
  disable colors with --no-color, HEROKU_LOGS_COLOR=0, or HEROKU_COLOR=0

EXAMPLES
  $ heroku logs
  2012-01-01T12:00:00+00:00 heroku[api]: Config add EXAMPLE by email@example.com
  2012-01-01T12:00:01+00:00 heroku[api]: Release v1 created by email@example.com
```

See code: @heroku-cli/plugin-run-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/run-v5/commands/logs.js)

heroku maintenance

display the current maintenance status of app

```
USAGE
  $ heroku maintenance

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku maintenance:off

take the app out of maintenance mode

```
USAGE
  $ heroku maintenance:off

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku maintenance:on

put the app into maintenance mode

```
USAGE
  $ heroku maintenance:on

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku members

list members of a team

```
USAGE
  $ heroku members

OPTIONS
  -r, --role=role  filter by role
  -t, --team=team  team to use
  --json           output in json format
  --pending        filter by pending team invitations
```

heroku members:add EMAIL

adds a user to a team

```
USAGE
  $ heroku members:add EMAIL

OPTIONS
  -r, --role=role  (required) member role (admin, collaborator, member, owner)
  -t, --team=team  team to use
```

heroku members:remove EMAIL

removes a user from a team

```
USAGE
  $ heroku members:remove EMAIL

OPTIONS
  -t, --team=team  team to use
```

heroku members:set EMAIL

sets a members role in a team

```
USAGE
  $ heroku members:set EMAIL

OPTIONS
  -r, --role=role  (required) member role (admin, collaborator, member, owner)
  -t, --team=team  team to use
```

heroku notifications

display notifications

```
USAGE
  $ heroku notifications

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

  --all                view all notifications (not just the ones for the current
                       app)

  --json               output in json format

  --read               show notifications already read
```

heroku orgs

list the teams that you are a member of

```
USAGE
  $ heroku orgs

OPTIONS
  --enterprise  filter by enterprise teams
  --json        output in json format
```

heroku orgs:open

open the team interface in a browser window

```
USAGE
  $ heroku orgs:open

OPTIONS
  -t, --team=team  team to use
```

heroku pg [DATABASE]

show database information

```
USAGE
  $ heroku pg [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:backups

list database backups

```
USAGE
  $ heroku pg:backups

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:backups:cancel [BACKUP_ID]

cancel an in-progress backup or restore (default newest)

```
USAGE
  $ heroku pg:backups:cancel [BACKUP_ID]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:backups:capture [DATABASE]

capture a new backup

```
USAGE
  $ heroku pg:backups:capture [DATABASE]

OPTIONS
  -a, --app=app                 (required) app to run command against
  -r, --remote=remote           git remote of app to use
  -v, --verbose
  --wait-interval=wait-interval
```

heroku pg:backups:delete BACKUP_ID

delete a backup

```
USAGE
  $ heroku pg:backups:delete BACKUP_ID

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use
```

heroku pg:backups:download [BACKUP_ID]

downloads database backup

```
USAGE
  $ heroku pg:backups:download [BACKUP_ID]

OPTIONS
  -a, --app=app        (required) app to run command against
  -o, --output=output  location to download to. Defaults to latest.dump
  -r, --remote=remote  git remote of app to use
```

heroku pg:backups:info [BACKUP_ID]

get information about a specific backup

```
USAGE
  $ heroku pg:backups:info [BACKUP_ID]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:backups:restore [BACKUP] [DATABASE]

restore a backup (default latest) to a database

```
USAGE
  $ heroku pg:backups:restore [BACKUP] [DATABASE]

OPTIONS
  -a, --app=app                  (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote            git remote of app to use
  -v, --verbose
  --wait-interval=wait-interval

DESCRIPTION
  defaults to saving the latest database to DATABASE_URL
```

heroku pg:backups:schedule [DATABASE]

schedule daily backups for given database

```
USAGE
  $ heroku pg:backups:schedule [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

  --at=at              (required) at a specific (24h) hour in the given
                       timezone. Defaults to UTC. --at '[HOUR]:00 [TIMEZONE]'
```

heroku pg:backups:schedules

list backup schedule

```
USAGE
  $ heroku pg:backups:schedules

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:backups:unschedule [DATABASE]

stop daily backups

```
USAGE
  $ heroku pg:backups:unschedule [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:backups:url [BACKUP_ID]

get secret but publicly accessible URL of a backup

```
USAGE
  $ heroku pg:backups:url [BACKUP_ID]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:bloat [DATABASE]

show table and index bloat in your database ordered by most wasteful

```
USAGE
  $ heroku pg:bloat [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:blocking [DATABASE]

display queries holding locks other queries are waiting to be released

```
USAGE
  $ heroku pg:blocking [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:connection-pooling:attach [DATABASE]

add an attachment to a database using connection pooling

```
USAGE
  $ heroku pg:connection-pooling:attach [DATABASE]

OPTIONS
  -a, --app=app                (required) app to run command against
  -n, --credential=credential  name of the credential within the database
  -r, --remote=remote          git remote of app to use
  --as=as                      name for add-on attachment

DESCRIPTION
  Example:

     heroku pg:connection-pooling:attach postgresql-something-12345 --credential
  cred-name
```

heroku pg:copy SOURCE TARGET

copy all data from source db to target

```
USAGE
  $ heroku pg:copy SOURCE TARGET

OPTIONS
  -a, --app=app                  (required) app to run command against
  -r, --remote=remote            git remote of app to use
  --confirm=confirm
  --verbose
  --wait-interval=wait-interval

DESCRIPTION
  at least one of the databases must be a Heroku PostgreSQL DB
```

heroku pg:credentials [DATABASE]

show information on credentials in the database

```
USAGE
  $ heroku pg:credentials [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --reset              DEPRECATED
```

heroku pg:credentials:create [DATABASE]

create credential within database

```
USAGE
  $ heroku pg:credentials:create [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -n, --name=name      (required) name of the new credential within the database
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Example:

       heroku pg:credentials:create postgresql-something-12345 --name
  new-cred-name
```

heroku pg:credentials:destroy [DATABASE]

destroy credential within database

```
USAGE
  $ heroku pg:credentials:destroy [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -n, --name=name        (required) unique identifier for the credential
  -r, --remote=remote    git remote of app to use

DESCRIPTION
  Example:

       heroku pg:credentials:destroy postgresql-transparent-56874 --name
  cred-name -a woodstock-production
```

heroku pg:credentials:repair-default [DATABASE]

repair the permissions of the default credential within database

```
USAGE
  $ heroku pg:credentials:repair-default [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use

DESCRIPTION
  Example:

       heroku pg:credentials:repair-default postgresql-something-12345
```

heroku pg:credentials:rotate [DATABASE]

rotate the database credentials

```
USAGE
  $ heroku pg:credentials:rotate [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm

  -n, --name=name        which credential to rotate (default credentials if not
                         specified)

  -r, --remote=remote    git remote of app to use

  --all                  rotate all credentials

  --force                forces rotating the targeted credentials
```

heroku pg:credentials:url [DATABASE]

show information on a database credential

```
USAGE
  $ heroku pg:credentials:url [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against

  -n, --name=name      which credential to show (default credentials if not
                       specified)

  -r, --remote=remote  git remote of app to use
```

heroku pg:diagnose [DATABASE|REPORT_ID]

run or view diagnostics report

```
USAGE
  $ heroku pg:diagnose [DATABASE|REPORT_ID]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  defaults to DATABASE_URL database if no DATABASE is specified
  if REPORT_ID is specified instead, a previous report is displayed
```

heroku pg:info [DATABASE]

show database information

```
USAGE
  $ heroku pg:info [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:kill PID [DATABASE]

kill a query

```
USAGE
  $ heroku pg:kill PID [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -f, --force
  -r, --remote=remote  git remote of app to use
```

heroku pg:killall [DATABASE]

terminates all connections for all credentials

```
USAGE
  $ heroku pg:killall [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:links [DATABASE]

lists all databases and information on link

```
USAGE
  $ heroku pg:links [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:links:create REMOTE DATABASE

create a link between data stores

```
USAGE
  $ heroku pg:links:create REMOTE DATABASE

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --as=as              name of link to create

DESCRIPTION
  Example:

       heroku pg:links:create HEROKU_REDIS_RED HEROKU_POSTGRESQL_CERULEAN
```

heroku pg:links:destroy DATABASE LINK

destroys a link between data stores

```
USAGE
  $ heroku pg:links:destroy DATABASE LINK

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use

DESCRIPTION
  Example:

       heroku pg:links:destroy HEROKU_POSTGRESQL_CERULEAN redis-symmetrical-100
```

heroku pg:locks [DATABASE]

display queries with active locks

```
USAGE
  $ heroku pg:locks [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -t, --truncate       truncates queries to 40 charaters
```

heroku pg:maintenance [DATABASE]

show current maintenance information

```
USAGE
  $ heroku pg:maintenance [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:maintenance:run [DATABASE]

start maintenance

```
USAGE
  $ heroku pg:maintenance:run [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -f, --force
  -r, --remote=remote  git remote of app to use
```

heroku pg:maintenance:window DATABASE WINDOW

set weekly maintenance window

```
USAGE
  $ heroku pg:maintenance:window DATABASE WINDOW

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  All times are in UTC.

  Example:

       heroku pg:maintenance:window postgres-slippery-100 "Sunday 06:00"
```

heroku pg:outliers [DATABASE]

show 10 queries that have longest execution time in aggregate

```
USAGE
  $ heroku pg:outliers [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -n, --num=num        the number of queries to display (default: 10)
  -r, --remote=remote  git remote of app to use
  -t, --truncate       truncate queries to 40 characters
  --reset              resets statistics gathered by pg_stat_statements
```

heroku pg:promote DATABASE

sets DATABASE as your DATABASE_URL

```
USAGE
  $ heroku pg:promote DATABASE

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:ps [DATABASE]

view active queries with execution time

```
USAGE
  $ heroku pg:ps [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -v, --verbose
```

heroku pg:psql [DATABASE]

open a psql shell to the database

```
USAGE
  $ heroku pg:psql [DATABASE]

OPTIONS
  -a, --app=app            (required) app to run command against
  -c, --command=command    SQL command to run
  -f, --file=file          SQL file to run
  -r, --remote=remote      git remote of app to use
  --credential=credential  credential to use
```

heroku pg:pull SOURCE TARGET

pull Heroku database into local or remote database

```
USAGE
  $ heroku pg:pull SOURCE TARGET

OPTIONS
  -a, --app=app                            (required) app to run command against
  -r, --remote=remote                      git remote of app to use

  --exclude-table-data=exclude-table-data  tables for which data should be
                                           excluded (use ';' to split multiple
                                           names)

DESCRIPTION
  Pull from SOURCE into TARGET.

  TARGET must be one of:
     * a database name (i.e. on a local PostgreSQL server)  => TARGET must not
  exist and will be created
     * a fully qualified URL to a local PostgreSQL server   => TARGET must not
  exist and will be created
     * a fully qualified URL to a remote PostgreSQL server  => TARGET must exist
  and be empty

  To delete a local database run `dropdb TARGET`
  To create an empty remote database, run `createdb` with connection
  command-line options (run `createdb --help` for details).

  Examples:

       # pull Heroku DB named postgresql-swimmingly-100 into local DB mylocaldb
  that must not exist
       $ heroku pg:pull postgresql-swimmingly-100 mylocaldb --app sushi

       # pull Heroku DB named postgresql-swimmingly-100 into empty remote DB at
  postgres://myhost/mydb
       $ heroku pg:pull postgresql-swimmingly-100 postgres://myhost/mydb --app
  sushi
```

heroku pg:push SOURCE TARGET

push local or remote into Heroku database

```
USAGE
  $ heroku pg:push SOURCE TARGET

OPTIONS
  -a, --app=app                           (required) app to run command against
  -r, --remote=remote                     git remote of app to use

  --exclude-table-data=exclude-table-data  tables for which data should be
                                          excluded (use ';' to split multiple
                                          names)

DESCRIPTION
  Push from SOURCE into TARGET. TARGET must be empty.

  To empty a Heroku database for push run `heroku pg:reset`

  SOURCE must be either the name of a database existing on your localhost or the
  fully qualified URL of a remote database.

  Examples:

       # push mylocaldb into a Heroku DB named postgresql-swimmingly-100
       $ heroku pg:push mylocaldb postgresql-swimmingly-100

       # push remote DB at postgres://myhost/mydb into a Heroku DB named
  postgresql-swimmingly-100
       $ heroku pg:push postgres://myhost/mydb postgresql-swimmingly-100
```

heroku pg:repoint [DATABASE]

changes which leader a follower is following

```
USAGE
  $ heroku pg:repoint [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use
  --follow=follow        leader database to follow

DESCRIPTION
  Example:

       heroku pg:repoint postgresql-transparent-56874 --follow
  postgresql-lucid-59103 -a woodstock-production
```

heroku pg:reset [DATABASE]

delete all data in DATABASE

```
USAGE
  $ heroku pg:reset [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use
```

heroku pg:settings [DATABASE]

show your current database settings

```
USAGE
  $ heroku pg:settings [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:settings:log-lock-waits [VALUE] [DATABASE]

Controls whether a log message is produced when a session waits longer than the deadlock_timeout to acquire a lock. deadlock_timeout is set to 1 second

```
USAGE
  $ heroku pg:settings:log-lock-waits [VALUE] [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Delays due to lock contention occur when multiple transactions are trying to
  access the same resource at the same time.
  Applications and their query patterns should try to avoid changes to many
  different tables within the same transaction.
```

heroku pg:settings:log-min-duration-statement [VALUE] [DATABASE]

The duration of each completed statement will be logged if the statement completes after the time specified by VALUE.

```
USAGE
  $ heroku pg:settings:log-min-duration-statement [VALUE] [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  VALUE needs to specified as a whole number, in milliseconds.
  Setting log_min_duration_statement to zero prints all statement durations and
  -1 will disable logging statement durations.
```

heroku pg:settings:log-statement [VALUE] [DATABASE]

log_statement controls which SQL statements are logged.

```
USAGE
  $ heroku pg:settings:log-statement [VALUE] [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Valid values for VALUE:
  none - No statements are logged
  ddl  - All data definition statements, such as CREATE, ALTER and DROP will be
  logged
  mod  - Includes all statements from ddl as well as data-modifying statements
  such as INSERT, UPDATE, DELETE, TRUNCATE, COPY
  all  - All statements are logged
```

heroku pg:unfollow DATABASE

stop a replica from following and make it a writeable database

```
USAGE
  $ heroku pg:unfollow DATABASE

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use
```

heroku pg:upgrade [DATABASE]

unfollow a database and upgrade it to the latest stable PostgreSQL version

```
USAGE
  $ heroku pg:upgrade [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use
  -v, --version=version  PostgreSQL version to upgrade to

DESCRIPTION
  to upgrade to another PostgreSQL version, use pg:copy instead
```

heroku pg:vacuum-stats [DATABASE]

show dead rows and whether an automatic vacuum is expected to be triggered

```
USAGE
  $ heroku pg:vacuum-stats [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku pg:wait [DATABASE]

blocks until database is available

```
USAGE
  $ heroku pg:wait [DATABASE]

OPTIONS
  -a, --app=app                  (required) app to run command against
  -r, --remote=remote            git remote of app to use
  --no-notify                    do not show OS notification

  --wait-interval=wait-interval  how frequently to poll in seconds (to avoid
                                 rate limiting)
```

heroku pipelines

list pipelines you have access to

```
USAGE
  $ heroku pipelines

OPTIONS
  --json  output in json format

EXAMPLES
  $ heroku pipelines
  === My Pipelines
  example
  sushi
```

heroku pipelines:add PIPELINE

add this app to a pipeline

```
USAGE
  $ heroku pipelines:add PIPELINE

ARGUMENTS
  PIPELINE  name of pipeline

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -s, --stage=stage    stage of first app in pipeline

DESCRIPTION
  The app and pipeline names must be specified.
  The stage of the app will be guessed based on its name if not specified.

EXAMPLES
  $ heroku pipelines:add example -a example-admin -s production
  Adding example-admin to example pipeline as production... done
```

heroku pipelines:connect [NAME]

connect a github repo to an existing pipeline

```
USAGE
  $ heroku pipelines:connect [NAME]

ARGUMENTS
  NAME  name of pipeline

OPTIONS
  -r, --repo=repo  (required) the GitHub repository to connect

EXAMPLES
  $ heroku pipelines:connect example -r githuborg/reponame
  Configuring pipeline... done
```

heroku pipelines:create [NAME]

create a new pipeline

```
USAGE
  $ heroku pipelines:create [NAME]

ARGUMENTS
  NAME  name of pipeline, defaults to basename of app

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -s, --stage=stage    stage of first app in pipeline
  -t, --team=team      team to use

DESCRIPTION
  An existing app must be specified as the first app in the pipeline.
  The pipeline name will be inferred from the app name if not specified.
  The stage of the app will be guessed based on its name if not specified.
  The pipeline owner will be the user creating the pipeline if not specified
  with -t for teams or -o for orgs.

EXAMPLES
  $ heroku pipelines:create -a example-staging
  ? Pipeline name: example
  ? Stage of example-staging: staging
  Creating example pipeline... done
  Adding example-staging to example pipeline as staging... done
```

heroku pipelines:destroy PIPELINE

destroy a pipeline

```
USAGE
  $ heroku pipelines:destroy PIPELINE

ARGUMENTS
  PIPELINE  name of pipeline

EXAMPLES
  $ heroku pipelines:destroy example
  Destroying example pipeline... done
```

heroku pipelines:diff

compares the latest release of this app to its downstream app(s)

```
USAGE
  $ heroku pipelines:diff

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLES
  $ heroku pipelines:diff --app murmuring-headland-14719
```

heroku pipelines:info PIPELINE

show list of apps in a pipeline

```
USAGE
  $ heroku pipelines:info PIPELINE

ARGUMENTS
  PIPELINE  pipeline to show

OPTIONS
  --json  output in json format

EXAMPLES
  $ heroku pipelines:info example
  === example
  owner: my-team (team)

  app name                     stage
  ──────────────────────────── ──────────
  ⬢ example-pr-16              review
  ⬢ example-pr-19              review
  ⬢ example-pr-23              review
  ⬢ example-staging            staging
  ⬢ example-staging-2          staging
  ⬢ example-production         production
```

heroku pipelines:list

list pipelines you have access to

```
USAGE
  $ heroku pipelines:list

OPTIONS
  --json  output in json format

EXAMPLES
  $ heroku pipelines
  === My Pipelines
  example
  sushi
```

heroku pipelines:open PIPELINE

open a pipeline in dashboard

```
USAGE
  $ heroku pipelines:open PIPELINE

ARGUMENTS
  PIPELINE  name of pipeline

EXAMPLES
  $ heroku pipelines:open example
```

heroku pipelines:promote

promote the latest release of this app to its downstream app(s)

```
USAGE
  $ heroku pipelines:promote

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -t, --to=to          comma separated list of apps to promote to

EXAMPLES
  $ heroku pipelines:promote -a example-staging
  Promoting example-staging to example (production)... done, v23
  Promoting example-staging to example-admin (production)... done, v54

  $ heroku pipelines:promote -a example-staging --to
  my-production-app1,my-production-app2
  Starting promotion to apps: my-production-app1,my-production-app2... done
  Waiting for promotion to complete... done
  Promotion successful
  my-production-app1: succeeded
  my-production-app2: succeeded
```

heroku pipelines:remove

remove this app from its pipeline

```
USAGE
  $ heroku pipelines:remove

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLES
  $ heroku pipelines:remove -a example-admin
  Removing example-admin... done
```

heroku pipelines:rename PIPELINE NAME

rename a pipeline

```
USAGE
  $ heroku pipelines:rename PIPELINE NAME

ARGUMENTS
  PIPELINE  current name of pipeline
  NAME      new name of pipeline

EXAMPLES
  $ heroku pipelines:rename example www
  Renaming example pipeline to www... done
```

heroku pipelines:setup [NAME] [REPO]

bootstrap a new pipeline with common settings and create a production and staging app (requires a fully formed app.json in the repo)

```
USAGE
  $ heroku pipelines:setup [NAME] [REPO]

ARGUMENTS
  NAME  name of pipeline
  REPO  a GitHub repository to connect the pipeline to

OPTIONS
  -t, --team=team  team to use
  -y, --yes        accept all default settings without prompting

EXAMPLES
  $ heroku pipelines:setup example githuborg/reponame -o example-org
  ? Automatically deploy the master branch to staging? Yes
  ? Wait for CI to pass before deploying the master branch to staging? Yes
  ? Enable review apps? Yes
  ? Automatically create review apps for every PR? Yes
  ? Automatically destroy idle review apps after 5 days? Yes
  ? Enable automatic Heroku CI test runs? Yes
  Creating pipeline... done
  Linking to repo... done
  Creating production and staging apps (⬢ example and ⬢ example-staging)
  Configuring pipeline... done
  View your new pipeline by running `heroku pipelines:open
  e5a55ffa-de3f-11e6-a245-3c15c2e6bc1e`
```

heroku pipelines:transfer OWNER

transfer ownership of a pipeline

```
USAGE
  $ heroku pipelines:transfer OWNER

ARGUMENTS
  OWNER  the owner to transfer the pipeline to

OPTIONS
  -c, --confirm=confirm
  -p, --pipeline=pipeline  (required) name of pipeline

EXAMPLES
  $ heroku pipelines:transfer me@example.com -p example
  === example

  app name              stage
  ────────────────────  ───────────
  ⬢ example-dev         development
  ⬢ example-staging     staging
  ⬢ example-prod        production

    ▸    This will transfer example and all of the listed apps to the
  me@example.com account
    ▸    to proceed, type example or re-run this command with --confirm example
  > example
  Transferring example pipeline to the me@example.com account... done

  $ heroku pipelines:transfer acme-widgets -p example
  === example

  app name              stage
  ────────────────────  ───────────
  ⬢ example-dev         development
  ⬢ example-staging     staging
  ⬢ example-prod        production

    ▸    This will transfer example and all of the listed apps to the
  acme-widgets team
    ▸    to proceed, type example or re-run this command with --confirm example
  > example

  Transferring example pipeline to the acme-widgets team... done
```

heroku plugins

list installed plugins

```
USAGE
  $ heroku plugins

OPTIONS
  --core  show core plugins

EXAMPLE
  $ heroku plugins
```

See code: @oclif/plugin-plugins (https://github.com/oclif/plugin-plugins/blob/v1.7.8/src/commands/plugins/index.ts)

heroku plugins:install PLUGIN...

installs a plugin into the CLI

```
USAGE
  $ heroku plugins:install PLUGIN...

ARGUMENTS
  PLUGIN  plugin to install

OPTIONS
  -f, --force    yarn install with force flag
  -h, --help     show CLI help
  -v, --verbose

DESCRIPTION
  Can be installed from npm or a git url.

  Installation of a user-installed plugin will override a core plugin.

  e.g. If you have a core plugin that has a 'hello' command, installing a
  user-installed plugin with a 'hello' command will override the core plugin
  implementation. This is useful if a user needs to update core plugin
  functionality in the CLI without the need to patch and update the whole CLI.

ALIASES
  $ heroku plugins:add

EXAMPLES
  $ heroku plugins:install myplugin
  $ heroku plugins:install https://github.com/someuser/someplugin
  $ heroku plugins:install someuser/someplugin
```

See code: @oclif/plugin-plugins (https://github.com/oclif/plugin-plugins/blob/v1.7.8/src/commands/plugins/install.ts)

heroku plugins:link PLUGIN

links a plugin into the CLI for development

```
USAGE
  $ heroku plugins:link PLUGIN

ARGUMENTS
  PATH  [default: .] path to plugin

OPTIONS
  -h, --help     show CLI help
  -v, --verbose

DESCRIPTION
  Installation of a linked plugin will override a user-installed or core plugin.

  e.g. If you have a user-installed or core plugin that has a 'hello' command,
  installing a linked plugin with a 'hello' command will override the
  user-installed or core plugin implementation. This is useful for development
  work.

EXAMPLE
  $ heroku plugins:link myplugin
```

See code: @oclif/plugin-plugins (https://github.com/oclif/plugin-plugins/blob/v1.7.8/src/commands/plugins/link.ts)

heroku plugins:uninstall PLUGIN...

removes a plugin from the CLI

```
USAGE
  $ heroku plugins:uninstall PLUGIN...

ARGUMENTS
  PLUGIN  plugin to uninstall

OPTIONS
  -h, --help     show CLI help
  -v, --verbose

ALIASES
  $ heroku plugins:unlink
  $ heroku plugins:remove
```

See code: @oclif/plugin-plugins (https://github.com/oclif/plugin-plugins/blob/v1.7.8/src/commands/plugins/uninstall.ts)

heroku plugins:update

update installed plugins

```
USAGE
  $ heroku plugins:update

OPTIONS
  -h, --help     show CLI help
  -v, --verbose
```

See code: @oclif/plugin-plugins (https://github.com/oclif/plugin-plugins/blob/v1.7.8/src/commands/plugins/update.ts)

heroku ps [TYPE [TYPE ...]]

list dynos for an app

```
USAGE
  $ heroku ps [TYPE [TYPE ...]]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --json               display as json

EXAMPLES
  $ heroku ps
  === run: one-off dyno
  run.1: up for 5m: bash

  === web: bundle exec thin start -p $PORT
  web.1: created for 30s

  $ heroku ps run # specifying types
  === run: one-off dyno
  run.1: up for 5m: bash
```

heroku ps:autoscale:disable

disable web dyno autoscaling

```
USAGE
  $ heroku ps:autoscale:disable

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

See code: @heroku-cli/plugin-ps (https://github.com/heroku/cli/blob/v7.24.0/packages/ps/src/commands/ps/autoscale/disable.ts)

heroku ps:autoscale:enable

enable web dyno autoscaling

```
USAGE
  $ heroku ps:autoscale:enable

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --max=max            (required) maximum number of dynos
  --min=min            (required) minimum number of dynos

  --notifications      receive email notifications when the max dyno limit is
                       reached

  --p95=p95            desired p95 response time
```

See code: @heroku-cli/plugin-ps (https://github.com/heroku/cli/blob/v7.24.0/packages/ps/src/commands/ps/autoscale/enable.ts)

heroku ps:copy FILE

Copy a file from a dyno to the local filesystem

```
USAGE
  $ heroku ps:copy FILE

OPTIONS
  -a, --app=app        (required) app to run command against
  -d, --dyno=dyno      specify the dyno to connect to
  -o, --output=output  the name of the output file
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Example:

       $ heroku ps:copy FILENAME --app murmuring-headland-14719
```

heroku ps:exec

Create an SSH session to a dyno

```
USAGE
  $ heroku ps:exec

OPTIONS
  -a, --app=app        (required) app to run command against
  -d, --dyno=dyno      specify the dyno to connect to
  -r, --remote=remote  git remote of app to use
  --ssh                use native ssh
  --status             lists the status of the SSH server in the dyno

DESCRIPTION
  Example:

       $ heroku ps:exec 'node -i' --app murmuring-headland-14719
```

heroku ps:forward PORT

Forward traffic on a local port to a dyno

```
USAGE
  $ heroku ps:forward PORT

OPTIONS
  -a, --app=app              (required) app to run command against
  -d, --dyno=dyno            specify the dyno to connect to
  -p, --localPort=localPort  the local port to use
  -r, --remote=remote        git remote of app to use

DESCRIPTION
  Example:

        $ heroku ps:forward 8080 --app murmuring-headland-14719
```

heroku ps:kill DYNO

stop app dyno

```
USAGE
  $ heroku ps:kill DYNO

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  stop app dyno or dyno type

EXAMPLES
  $ heroku ps:stop run.1828
  Stopping run.1828 dyno... done

  $ heroku ps:stop run
  Stopping run dynos... done
```

heroku ps:resize

manage dyno sizes

```
USAGE
  $ heroku ps:resize

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Called with no arguments shows the current dyno size.

  Called with one argument sets the size.
  Where SIZE is one of free|hobby|standard-1x|standard-2x|performance

  Called with 1..n TYPE=SIZE arguments sets the quantity per type.
```

heroku ps:restart [DYNO]

restart app dynos

```
USAGE
  $ heroku ps:restart [DYNO]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  if DYNO is not specified, restarts all dynos on app

EXAMPLES
  $ heroku ps:restart web.1
  Restarting web.1 dyno... done

  $ heroku ps:restart web
  Restarting web dynos... done

  $ heroku ps:restart
  Restarting dynos... done
```

heroku ps:scale

scale dyno quantity up or down

```
USAGE
  $ heroku ps:scale

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Appending a size (eg. web=2:Standard-2X) allows simultaneous scaling and
  resizing.

  Omitting any arguments will display the app's current dyno formation, in a
  format suitable for passing back into ps:scale.

EXAMPLES
  $ heroku ps:scale web=3:Standard-2X worker+1
  Scaling dynos... done, now running web at 3:Standard-2X, worker at
  1:Standard-1X.

  $ heroku ps:scale
  web=3:Standard-2X worker=1:Standard-1X
```

heroku ps:socks

Launch a SOCKS proxy into a dyno

```
USAGE
  $ heroku ps:socks

OPTIONS
  -a, --app=app        (required) app to run command against
  -d, --dyno=dyno      specify the dyno to connect to
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Example:

        $ heroku ps:socks --app murmuring-headland-14719
        Establishing credentials... done
        SOCKSv5 proxy server started on port 1080
        Use CTRL+C to stop the proxy
```

heroku ps:stop DYNO

stop app dyno

```
USAGE
  $ heroku ps:stop DYNO

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  stop app dyno or dyno type

EXAMPLES
  $ heroku ps:stop run.1828
  Stopping run.1828 dyno... done

  $ heroku ps:stop run
  Stopping run dynos... done
```

heroku ps:type

manage dyno sizes

```
USAGE
  $ heroku ps:type

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Called with no arguments shows the current dyno size.

  Called with one argument sets the size.
  Where SIZE is one of free|hobby|standard-1x|standard-2x|performance

  Called with 1..n TYPE=SIZE arguments sets the quantity per type.
```

heroku ps:wait

wait for all dynos to be running latest version after a release

```
USAGE
  $ heroku ps:wait

OPTIONS
  -R, --with-run                    whether to wait for one-off run dynos
  -a, --app=app                     (required) app to run command against
  -r, --remote=remote               git remote of app to use
  -t, --type=type                   wait for one specific dyno type

  -w, --wait-interval=wait-interval  [default: 10] how frequently to poll in
                                    seconds (to avoid hitting Heroku API rate
                                    limits)
```

See code: @heroku-cli/plugin-ps (https://github.com/heroku/cli/blob/v7.24.0/packages/ps/src/commands/ps/wait.ts)

heroku psql [DATABASE]

open a psql shell to the database

```
USAGE
  $ heroku psql [DATABASE]

OPTIONS
  -a, --app=app            (required) app to run command against
  -c, --command=command    SQL command to run
  -f, --file=file          SQL file to run
  -r, --remote=remote      git remote of app to use
  --credential=credential  credential to use
```

heroku redis [DATABASE]

gets information about redis

```
USAGE
  $ heroku redis [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku redis:cli [DATABASE]

opens a redis prompt

```
USAGE
  $ heroku redis:cli [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -c, --confirm=confirm
  -r, --remote=remote    git remote of app to use
```

heroku redis:credentials [DATABASE]

display credentials information

```
USAGE
  $ heroku redis:credentials [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  --reset              reset credentials
```

heroku redis:info [DATABASE]

gets information about redis

```
USAGE
  $ heroku redis:info [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku redis:maintenance [DATABASE]

manage maintenance windows

```
USAGE
  $ heroku redis:maintenance [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against

  -f, --force          start maintenance without entering application
                       maintenance mode

  -r, --remote=remote  git remote of app to use

  -w, --window=window  set weekly UTC maintenance window

  --run                start maintenance

DESCRIPTION
  Set or change the maintenance window for your Redis instance
```

heroku redis:maxmemory [DATABASE]

set the key eviction policy

```
USAGE
  $ heroku redis:maxmemory [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -p, --policy=policy  (required) set policy name
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  Set the key eviction policy when instance reaches its storage limit. Available
  policies for key eviction include:

       noeviction      # returns errors when memory limit is reached
       allkeys-lfu     # removes less frequently used keys first
       volatile-lfu    # removes less frequently used keys first that have an
  expiry set
       allkeys-lru     # removes less recently used keys first
       volatile-lru    # removes less recently used keys first that have an
  expiry set
       allkeys-random  # evicts random keys
       volatile-random # evicts random keys but only those that have an expiry
  set
       volatile-ttl    # only evicts keys with an expiry set and a short TTL
```

heroku redis:promote DATABASE

sets DATABASE as your REDIS_URL

```
USAGE
  $ heroku redis:promote DATABASE

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku redis:timeout [DATABASE]

set the number of seconds to wait before killing idle connections

```
USAGE
  $ heroku redis:timeout [DATABASE]

OPTIONS
  -a, --app=app          (required) app to run command against
  -r, --remote=remote    git remote of app to use
  -s, --seconds=seconds  set timeout value

DESCRIPTION
  Sets the number of seconds to wait before killing idle connections. A value of
  zero means that connections will not be closed.
```

heroku redis:wait [DATABASE]

wait for Redis instance to be available

```
USAGE
  $ heroku redis:wait [DATABASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku regions

list available regions for deployment

```
USAGE
  $ heroku regions

OPTIONS
  --common   show regions for common runtime
  --json     output in json format
  --private  show regions for private spaces
```

See code: @heroku-cli/plugin-ps (https://github.com/heroku/cli/blob/v7.24.0/packages/ps/src/commands/regions.ts)

heroku releases

display the releases for an app

```
USAGE
  $ heroku releases

OPTIONS
  -a, --app=app        (required) app to run command against
  -n, --num=num        number of releases to show
  -r, --remote=remote  git remote of app to use
  --json               output releases in json format

EXAMPLES
  $ heroku releases
  === example Releases
  v1 Config add FOO_BAR email@example.com 2015/11/17 17:37:41 (~ 1h ago)
  v2 Config add BAR_BAZ email@example.com 2015/11/17 17:37:41 (~ 1h ago)
  v3 Config add BAZ_QUX email@example.com 2015/11/17 17:37:41 (~ 1h ago)
```

heroku releases:info [RELEASE]

view detailed information for a release

```
USAGE
  $ heroku releases:info [RELEASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
  -s, --shell          output in shell format
  --json               output in json format
```

heroku releases:output [RELEASE]

View the release command output

```
USAGE
  $ heroku releases:output [RELEASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku releases:rollback [RELEASE]

rollback to a previous release

```
USAGE
  $ heroku releases:rollback [RELEASE]

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use

DESCRIPTION
  If RELEASE is not specified, it will rollback one release
```

heroku reviewapps:disable

disable review apps or settings on an existing pipeline

```
USAGE
  $ heroku reviewapps:disable

OPTIONS
  -a, --app=app            (required) parent app used by review apps
  -p, --pipeline=pipeline  (required) name of pipeline
  -r, --remote=remote      git remote of parent app used by review apps
  --autodeploy             disable autodeployments
  --autodestroy            disable automatically destroying review apps

EXAMPLES
  $ heroku reviewapps:disable -p mypipeline -a myapp --autodeploy
  Disabling auto deployment ...
  Configuring pipeline... done
```

heroku reviewapps:enable

enable review apps and/or settings on an existing pipeline

```
USAGE
  $ heroku reviewapps:enable

OPTIONS
  -a, --app=app            (required) parent app used by review apps
  -p, --pipeline=pipeline  (required) name of pipeline
  --autodeploy             autodeploy the review app
  --autodestroy            autodestroy the review app

EXAMPLE
  $ heroku reviewapps:enable -p mypipeline -a myapp --autodeploy --autodestroy
```

See code: @heroku-cli/plugin-pipelines (https://github.com/heroku/heroku-cli-plugin-pipelines/blob/v7.27.0/src/commands/reviewapps/enable.ts)

heroku run

run a one-off process inside a heroku dyno

```
USAGE
  $ heroku run

OPTIONS
  -a, --app=app        (required) app to run command against

  -e, --env=env        environment variables to set (use ';' to split multiple
                       vars)

  -r, --remote=remote  git remote of app to use

  -s, --size=size      dyno size

  -x, --exit-code      passthrough the exit code of the remote command

  --no-notify          disables notification when dyno is up (alternatively use
                       HEROKU_NOTIFICATIONS=0)

  --no-tty             force the command to not run in a tty

  --type=type          process type

DESCRIPTION
  Shows a notification if the dyno takes more than 20 seconds to start.

EXAMPLES
  $ heroku run bash
  Running bash on app.... up, run.1
  ~ $

  $ heroku run -s hobby -- myscript.sh -a arg1 -s arg2
  Running myscript.sh -a arg1 -s arg2 on app.... up, run.1
```

See code: @heroku-cli/plugin-run-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/run-v5/commands/run.js)

heroku run:detached

run a detached dyno, where output is sent to your logs

```
USAGE
  $ heroku run:detached

OPTIONS
  -a, --app=app        (required) app to run command against

  -e, --env=env        environment variables to set (use ';' to split multiple
                       vars)

  -r, --remote=remote  git remote of app to use

  -s, --size=size      dyno size

  -t, --tail           stream logs from the dyno

  --type=type          process type

EXAMPLES
  $ heroku run:detached ls
  Running ls on app [detached]... up, run.1
  Run heroku logs -a app -p run.1 to view the output.
```

See code: @heroku-cli/plugin-run-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/run-v5/commands/run/detached.js)

heroku sessions

list your OAuth sessions

```
USAGE
  $ heroku sessions

OPTIONS
  -j, --json  output in json format
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/sessions/index.js)

heroku sessions:destroy ID

delete (logout) OAuth session by ID

```
USAGE
  $ heroku sessions:destroy ID
```

See code: @heroku-cli/plugin-oauth-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/oauth-v5/lib/commands/sessions/destroy.js)

heroku spaces

list available spaces

```
USAGE
  $ heroku spaces

OPTIONS
  -t, --team=team  team to use
  --json           output in json format
```

heroku spaces:create

create a new space

```
USAGE
  $ heroku spaces:create

OPTIONS
  -s, --space=space      name of space to create
  -t, --team=team        team to use
  --cidr=cidr            RFC-1918 CIDR the space will use

  --data-cidr=data-cidr  RFC-1918 CIDR used by Heroku Data resources for the
                         space

  --region=region        region name

DESCRIPTION
  Example:

       $ heroku spaces:create --space my-space --team my-team --region oregon
       Creating space my-space in team my-team... done
       === my-space
       ID:         e7b99e37-69b3-4475-ad47-a5cc5d75fd9f
       Team:       my-team
       Region:     oregon
       CIDR:       10.0.0.0/16
       Data CIDR:  172.23.0.0/20
       State:      allocating
       Created at: 2016-01-06T03:23:13Z
```

heroku spaces:destroy

destroy a space

```
USAGE
  $ heroku spaces:destroy

OPTIONS
  -s, --space=space  space to destroy
  --confirm=confirm  set to space name to bypass confirm prompt

DESCRIPTION
  Example:

       $ heroku spaces:destroy --space my-space
       Destroying my-space... done
```

heroku spaces:info

show info about a space

```
USAGE
  $ heroku spaces:info

OPTIONS
  -s, --space=space  space to get info of
  --json             output in json format
```

heroku spaces:peering:info

display the information necessary to initiate a peering connection

```
USAGE
  $ heroku spaces:peering:info

OPTIONS
  -s, --space=space  space to get peering info from
  --json             output in json format

DESCRIPTION
  Example:

       $ heroku spaces:peering:info example-space
       === example-space Peering Info
       AWS Account ID:    012345678910
       AWS Region:        us-west-2
       AWS VPC ID:        vpc-baadf00d
       AWS VPC CIDR:      10.0.0.0/16
       Space CIDRs:       10.0.128.0/20, 10.0.144.0/20
       Unavailable CIDRs: 10.1.0.0/16

  You will use the information provied by this command to establish a peering
  connection request from your AWS VPC to your private space.

  To start the peering process, go into your AWS console for the VPC you would
  like peered with your Private Space,
  navigate to the VPC service, choose the "Peering Connections" option and click
  the "Create peering connection" button.

  - The AWS Account ID and VPC ID are necessary for the AWS VPC Peering
  connection wizard.
  - You will also need to configure your VPC route table to route the Dyno CIDRs
  through the peering connection.

  Once you've established the peering connection request, you can use the
  spaces:peerings:accept command to accept and
  configure the peering connection for the space.
```

heroku spaces:peerings

list peering connections for a space

```
USAGE
  $ heroku spaces:peerings

OPTIONS
  -s, --space=space  space to get peer list from
  --json             output in json format
```

heroku spaces:peerings:accept

accepts a pending peering request for a private space

```
USAGE
  $ heroku spaces:peerings:accept

OPTIONS
  -p, --pcxid=pcxid  PCX ID of a pending peering
  -s, --space=space  space to get peering info from

DESCRIPTION
  Example:

       $ heroku spaces:peerings:accept pcx-4bd27022 --space example-space
       Accepting and configuring peering connection pcx-4bd27022
```

heroku spaces:peerings:destroy

destroys an active peering connection in a private space

```
USAGE
  $ heroku spaces:peerings:destroy

OPTIONS
  -p, --pcxid=pcxid  PCX ID of a pending peering
  -s, --space=space  space to get peering info from
  --confirm=confirm  set to PCX ID to bypass confirm prompt

DESCRIPTION
  Example:

       $ heroku spaces:peerings:destroy pcx-4bd27022 --confirm pcx-4bd27022
  --space example-space
       Tearing down peering connection pcx-4bd27022
```

heroku spaces:ps

list dynos for a space

```
USAGE
  $ heroku spaces:ps

OPTIONS
  -s, --space=space  space to get dynos of
  --json             output in json format
```

heroku spaces:rename

renames a space

```
USAGE
  $ heroku spaces:rename

OPTIONS
  --from=from  (required) current name of space
  --to=to      (required) desired name of space

DESCRIPTION
  Example:

       $ heroku spaces:rename --from old-space-name --to new-space-name
       Renaming space old-space-name to new-space-name... done
```

heroku spaces:topology

show space topology

```
USAGE
  $ heroku spaces:topology

OPTIONS
  -s, --space=space  space to get topology of
  --json             output in json format
```

heroku spaces:transfer

transfer a space to another team

```
USAGE
  $ heroku spaces:transfer

OPTIONS
  --space=space  (required) name of space
  --team=team    (required) desired owner of space

DESCRIPTION
  Example:

       $ heroku spaces:transfer --space=space-name --team=team-name
       Transferring space-name to team-name... done
```

heroku spaces:vpn:config

display the configuration information for VPN

```
USAGE
  $ heroku spaces:vpn:config

OPTIONS
  -n, --name=name    name or id of the VPN connection to retrieve config from
  -s, --space=space  space the VPN connection belongs to
  --json             output in json format

DESCRIPTION
  Example:

       $ heroku spaces:vpn:config --space my-space vpn-connection-name
       === vpn-connection-name VPN Tunnels
       VPN Tunnel  Customer Gateway  VPN Gateway     Pre-shared Key  Routable
  Subnets  IKE Version
       __________  _________________  ______________  ______________
  _________________  ____________
       Tunnel 1    104.196.121.200   35.171.237.136  abcdef12345     10.0.0.0/16
        1
       Tunnel 2    104.196.121.200   52.44.7.216     fedcba54321     10.0.0.0/16
        1

  You will use the information provided by this command to establish a Private
  Space VPN Connection.

  - You must configure your VPN Gateway to use both Tunnels provided by Heroku
  - The VPN Gateway values are the IP addresses of the Private Space Tunnels
  - The Customer Gateway value is the Public IP of your VPN Gateway
  - The VPN Gateway must use the IKE Version shown and the Pre-shared Keys as
  the authentication method
```

heroku spaces:vpn:connect

create VPN

```
USAGE
  $ heroku spaces:vpn:connect

OPTIONS
  -c, --cidrs=cidrs  a list of routable CIDRs separated by commas
  -i, --ip=ip        public IP of customer gateway
  -n, --name=name    VPN name
  -s, --space=space  space name

DESCRIPTION
  Private Spaces can be connected to another private network via an IPSec VPN
  connection allowing dynos to connect to hosts on your private networks and
  vice versa.
  The connection is established over the public Internet but all traffic is
  encrypted using IPSec.

EXAMPLES
  $ heroku spaces:vpn:connect --name office --ip 35.161.69.30 --cidrs
  172.16.0.0/16,10.0.0.0/24 --space my-space
       Creating VPN Connection in space my-space... done
       ▸    Use spaces:vpn:wait to track allocation.
```

heroku spaces:vpn:connections

list the VPN Connections for a space

```
USAGE
  $ heroku spaces:vpn:connections

OPTIONS
  -s, --space=space  space to get VPN connections from
  --json             output in json format

DESCRIPTION
  Example:

     $ heroku spaces:vpn:connections --space my-space
     === my-space VPN Connections
     Name    Status  Tunnels
     ______  ______  _______
     office  active  UP/UP
```

heroku spaces:vpn:destroy

destroys VPN in a private space

```
USAGE
  $ heroku spaces:vpn:destroy

OPTIONS
  -n, --name=name    name or id of the VPN connection to retrieve config from
  -s, --space=space  space to get peering info from
  --confirm=confirm  set to VPN connection name to bypass confirm prompt

DESCRIPTION
  Example:

       $ heroku spaces:vpn:destroy --space example-space vpn-connection-name
  --confirm vpn-connection-name
       Tearing down VPN Connection vpn-connection-name in space example-space
```

heroku spaces:vpn:info

display the information for VPN

```
USAGE
  $ heroku spaces:vpn:info

OPTIONS
  -n, --name=name    name or id of the VPN connection to get info from
  -s, --space=space  space the vpn connection belongs to
  --json             output in json format

DESCRIPTION
  Example:

       $ heroku spaces:vpn:info --space my-space vpn-connection-name
       === vpn-connection-name VPN Tunnel Info
       Name:           vpn-connection-name
       ID:             123456789012
       Public IP:      35.161.69.30
       Routable CIDRs: 172.16.0.0/16
       Status:         failed
       Status Message: supplied CIDR block already in use
       === my-space Tunnel Info
       VPN Tunnel  IP Address     Status  Status Last Changed   Details
       ___________ ______________ ______  ____________________  ______________
       Tunnel 1    52.44.146.197  UP      2016-10-25T22:09:05Z  status message
       Tunnel 2    52.44.146.197  UP      2016-10-25T22:09:05Z  status message
```

heroku spaces:vpn:wait

wait for VPN Connection to be created

```
USAGE
  $ heroku spaces:vpn:wait

OPTIONS
  -i, --interval=interval  seconds to wait between poll intervals
  -n, --name=name          name or id of the vpn connection to wait for
  -s, --space=space        space the vpn connection belongs to
  -t, --timeout=timeout    maximum number of seconds to wait
  --json                   output in json format
```

heroku spaces:wait

wait for a space to be created

```
USAGE
  $ heroku spaces:wait

OPTIONS
  -i, --interval=interval  seconds to wait between poll intervals
  -s, --space=space        space to get info of
  -t, --timeout=timeout    maximum number of seconds to wait
  --json                   output in json format
```

heroku status

display current status of the Heroku platform

```
USAGE
  $ heroku status

OPTIONS
  --json  output in json format
```

See code: @heroku-cli/plugin-status (https://github.com/heroku/cli/blob/v7.24.0/packages/status/src/commands/status.ts)

heroku teams

list the teams that you are a member of

```
USAGE
  $ heroku teams

OPTIONS
  --json  output in json format

DESCRIPTION
  Use heroku members:* to manage team members.
```

heroku trusted-ips

list trusted IP ranges for a space

```
USAGE
  $ heroku trusted-ips

OPTIONS
  -s, --space=space  space to get inbound rules from
  --json             output in json format

DESCRIPTION
  Trusted IP ranges are only available on Private Spaces.

  The space name is a required parameter. Newly created spaces will have
  0.0.0.0/0 set by default
  allowing all traffic to applications in the space. More than one CIDR block
  can be provided at
  a time to the commands listed below. For example 1.2.3.4/20 and 5.6.7.8/20 can
  be added with:
```

heroku trusted-ips:add SOURCE

Add one range to the list of trusted IP ranges

```
USAGE
  $ heroku trusted-ips:add SOURCE

OPTIONS
  -s, --space=space  space to add rule to
  --confirm=confirm  set to space name to bypass confirm prompt

DESCRIPTION
  Uses CIDR notation.

  Example:

       $ heroku trusted-ips:add --space my-space 192.168.2.0/24
       Added 192.168.0.1/24 to trusted IP ranges on my-space
```

heroku trusted-ips:remove SOURCE

Remove a range from the list of trusted IP ranges

```
USAGE
  $ heroku trusted-ips:remove SOURCE

OPTIONS
  --confirm=confirm  set to space name to bypass confirm prompt
  --space=space      (required) space to remove rule from

DESCRIPTION
  Uses CIDR notation.

  Example:

       $ heroku trusted-ips:remove --space my-space 192.168.2.0/24
       Removed 192.168.2.0/24 from trusted IP ranges on my-space
```

heroku unlock

unlock an app so any team member can join

```
USAGE
  $ heroku unlock

OPTIONS
  -a, --app=app        (required) app to run command against
  -r, --remote=remote  git remote of app to use
```

heroku update [CHANNEL]

update the heroku CLI

```
USAGE
  $ heroku update [CHANNEL]
```

See code: @oclif/plugin-update (https://github.com/oclif/plugin-update/blob/v1.3.9/src/commands/update.ts)

heroku webhooks

list webhooks on an app

```
USAGE
  $ heroku webhooks

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLE
  $ heroku webhooks
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/index.js)

heroku webhooks:add

add a webhook to an app

```
USAGE
  $ heroku webhooks:add

OPTIONS
  -a, --app=app                      app to run command against

  -i, --include=include              (required) comma delimited event types your
                                     server will receive

  -l, --level=level                  (required) notify does not retry, sync will
                                     retry until successful or timeout

  -r, --remote=remote                git remote of app to use

  -s, --secret=secret                value to sign delivery with in
                                     Heroku-Webhook-Hmac-SHA256 header

  -t, --authorization=authorization  authoriation header to send with webhooks

  -u, --url=url                      (required) URL for receiver

EXAMPLE
  $ heroku webhooks:add -i api:dyno -l notify -u https://example.com/hooks
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/add.js)

heroku webhooks:deliveries

list webhook deliveries on an app

```
USAGE
  $ heroku webhooks:deliveries

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use
  -s, --status=status  filter deliveries by status

EXAMPLE
  $ heroku webhooks:deliveries
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/deliveries/index.js)

heroku webhooks:deliveries:info [ID]

info for a webhook event on an app

```
USAGE
  $ heroku webhooks:deliveries:info [ID]

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLE
  $ heroku webhooks:deliveries:info 99999999-9999-9999-9999-999999999999
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/deliveries/info.js)

heroku webhooks:events

list webhook events on an app

```
USAGE
  $ heroku webhooks:events

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLE
  $ heroku webhooks:events
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/events/index.js)

heroku webhooks:events:info [ID]

info for a webhook event on an app

```
USAGE
  $ heroku webhooks:events:info [ID]

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLE
  $ heroku webhooks:events:info 99999999-9999-9999-9999-999999999999
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/events/info.js)

heroku webhooks:info [ID]

info for a webhook on an app

```
USAGE
  $ heroku webhooks:info [ID]

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLE
  $ heroku webhooks:info 99999999-9999-9999-9999-999999999999
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/info.js)

heroku webhooks:remove [ID]

removes a webhook from an app

```
USAGE
  $ heroku webhooks:remove [ID]

ARGUMENTS
  ID  id of webhook to remove

OPTIONS
  -a, --app=app        app to run command against
  -r, --remote=remote  git remote of app to use

EXAMPLE
  $ heroku webhooks:remove 99999999-9999-9999-9999-999999999999
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/remove.js)

heroku webhooks:update [ID]

updates a webhook in an app

```
USAGE
  $ heroku webhooks:update [ID]

OPTIONS
  -a, --app=app                      app to run command against

  -i, --include=include              (required) comma delimited event types your
                                     server will receive

  -l, --level=level                  (required) notify does not retry, sync will
                                     retry until successful or timeout

  -r, --remote=remote                git remote of app to use

  -s, --secret=secret                value to sign delivery with in
                                     Heroku-Webhook-Hmac-SHA256 header

  -t, --authorization=authorization  authoriation header to send with webhooks

  -u, --url=url                      (required) URL for receiver

EXAMPLE
  $ heroku webhooks:update 99999999-9999-9999-9999-999999999999 -i dyno -l
  notify -s 09928c40bf1b191b645174a19f7053d16a180da37332e719ef0998f4c0a2 -u
  https://example.com/hooks
```

See code: @heroku-cli/plugin-webhooks-v5 (https://github.com/heroku/cli/blob/v7.24.0/packages/webhooks-v5/commands/webhooks/update.js)

heroku which COMMAND

show which plugin a command is in

```
USAGE
  $ heroku which COMMAND
```

See code: @oclif/plugin-which (https://github.com/oclif/plugin-which/blob/v1.0.3/src/commands/which.ts)