Pipelines

Last updated 31 October 2019

Table of Contents

A **pipeline** is a group of Heroku apps that share the same codebase. Each app in a pipeline represents one of the following stages in a continuous delivery workflow (http://en.wikipedia.org/wiki/Continuous_delivery):

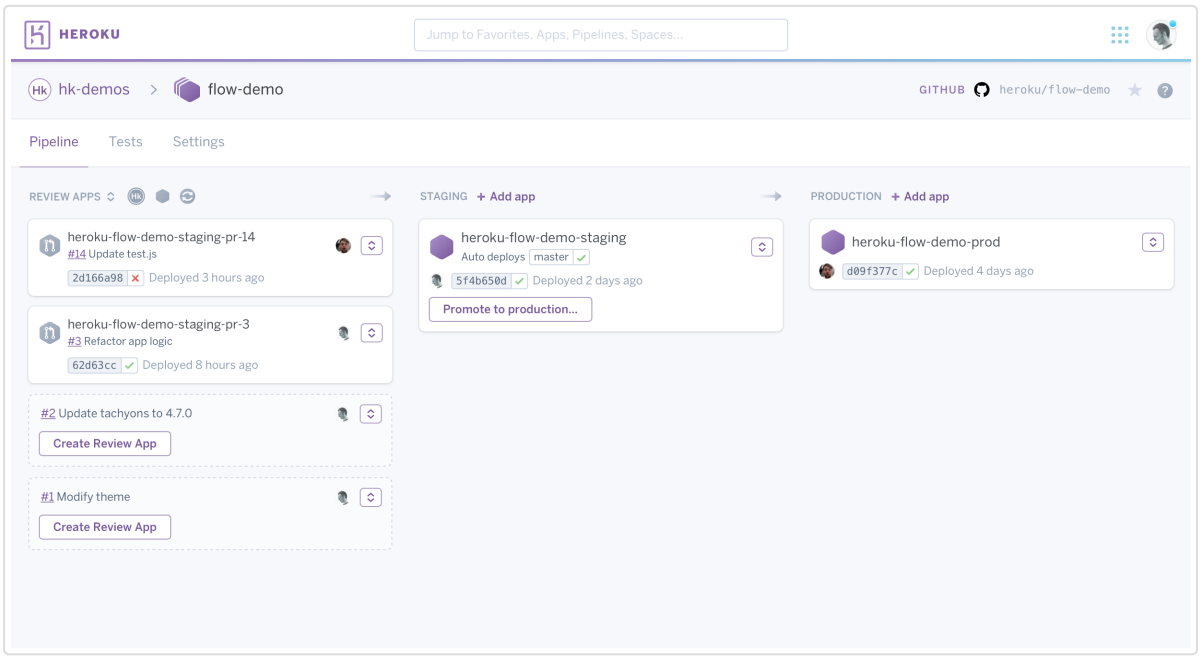
- Development
- Review
- Staging
- Production

Pipelines are extremely useful for managing multiple environments for your app (https://devcenter.heroku.com/articles/multiple-environments). A common pipeline workflow has the following steps:

1. A developer creates a pull request to make a change to the codebase.
2. Heroku automatically creates a review app (https://devcenter.heroku.com/articles/github-integration-review-apps) for the pull request, allowing developers to test the change.
3. When the change is ready, it's merged into the codebase's master branch.
4. The master branch is automatically deployed (https://devcenter.heroku.com/articles/github-integration) to the pipeline's staging app for further testing.

5. When the change is ready, a developer promotes the staging app to production, making it available to the app's end users.

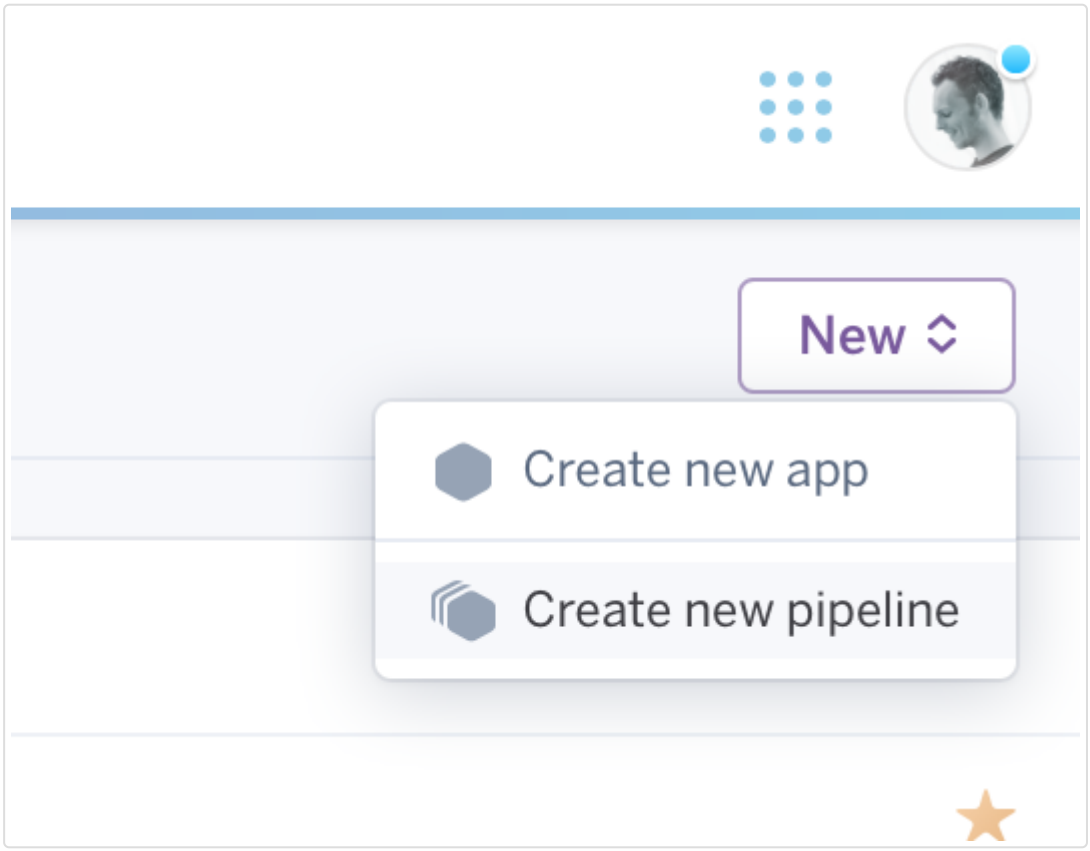
A pipeline's overview page illustrates the stages of this flow and provides meta-information about the status of each stage. For example, you can see if your production app is running different code than staging.



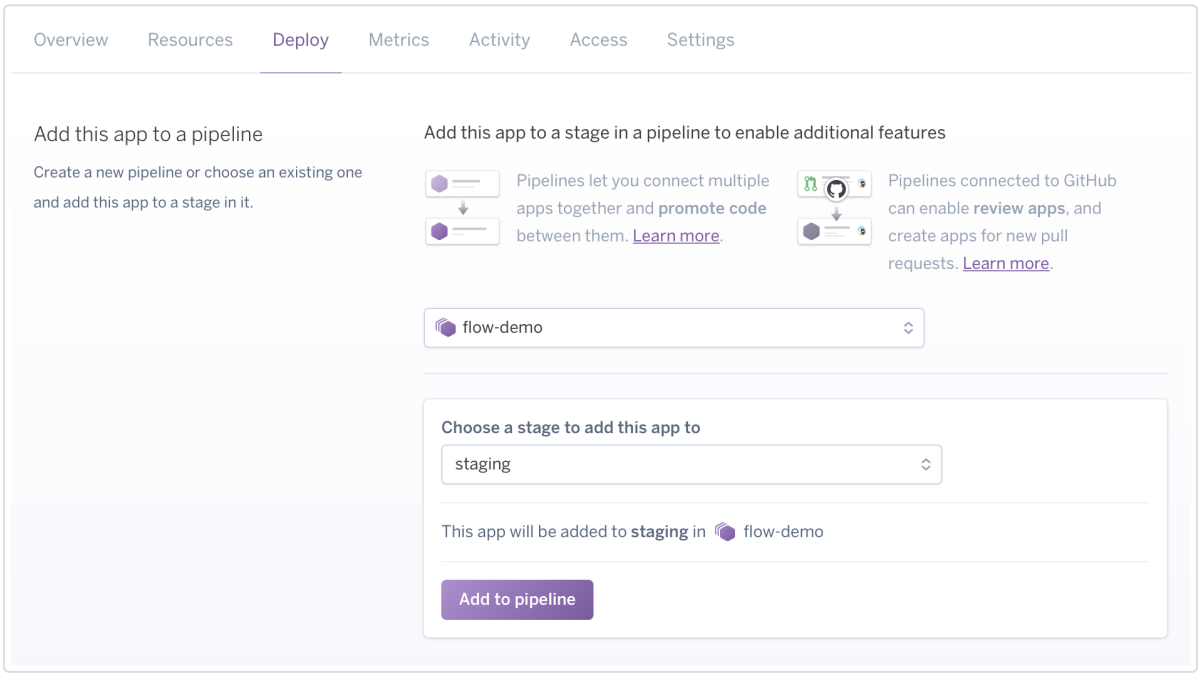
Creating pipelines

From the Heroku Dashboard

Click the `New` button in the top right of your app list and select `Create new pipeline`:



Alternatively, you can navigate to an app's **`Deploy`** tab and create a new pipeline to include that app.



From the Heroku CLI

You can use the `pipelines:create` command to create a pipeline from the CLI. Note that the command must specify an existing app that you want to add to the pipeline.

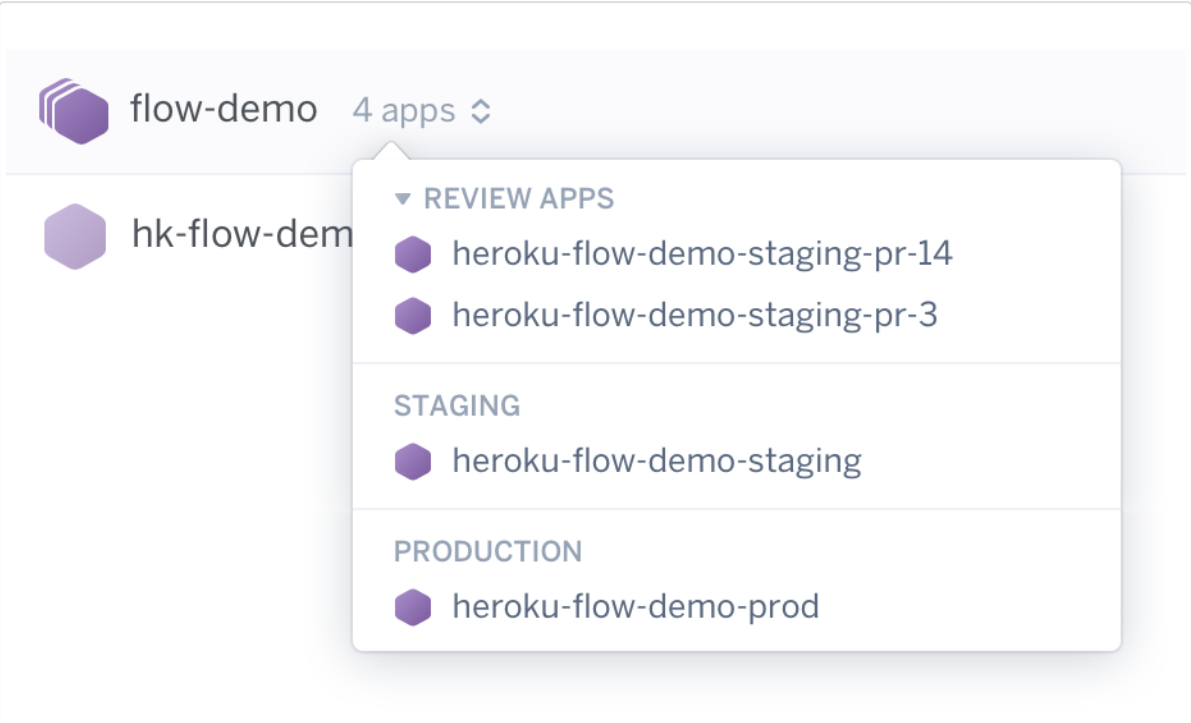
```
$ heroku pipelines:create -a example-app
? Pipeline name: example-pipeline
? Stage of example-app: production
Creating example-pipeline pipeline... done
Adding example-app to example-pipeline pipeline as production... done
```

The CLI prompts you to specify a name for the pipeline and a stage for the app you're adding to it (`development`, `staging`, or `production`).

Use `heroku help pipelines:create` to see a full list of options for this command.

Accessing pipelines

Apps in a pipeline do not appear as individual apps in the Heroku Dashboard. Instead, they appear as part of a single pipeline entry with a drop-down to view individual apps:



Adding apps to a pipeline

A Heroku app can belong to exactly one stage of exactly one pipeline.

From the Heroku Dashboard

From your pipeline's overview page, click `+ Add app` next to a deployment stage to add an existing application to that stage of the pipeline.

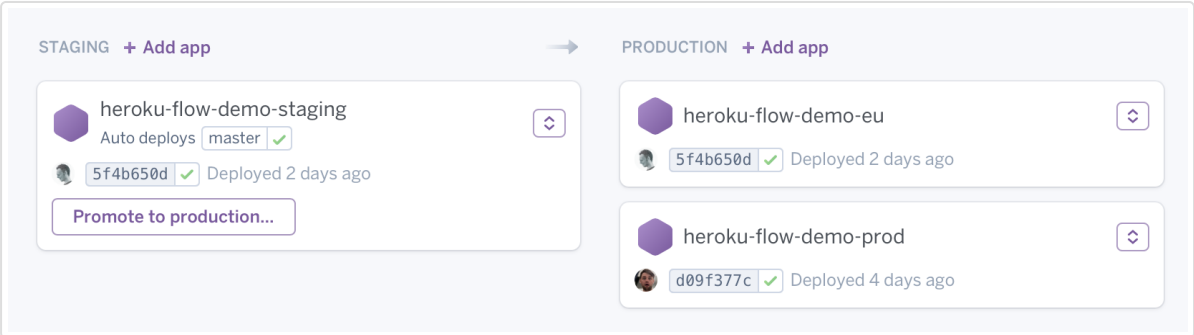
From the Heroku CLI

Add an app to your pipeline with the `heroku pipelines:add` command:

```
$ heroku pipelines:add example-pipeline -a example-staging-app
? Stage of example-staging: staging
Adding example-staging to example pipeline as staging... done
```

Multiple apps in a pipeline stage

Pipeline stages can include more than one app. For example, the production stage might have the main production app and an admin app running the same version of code, but with different configurations.



Deployment with pipelines

Pipelines let you define how your deployed code flows from one environment to the next. For example, you can deploy code to your staging app (which builds it into a slug (https://devcenter.heroku.com/articles/slug-compiler)) and later **promote** that same slug to production. This promotion flow ensures that production contains the exact same code that you tested in staging, and it's also much faster than rebuilding the slug.

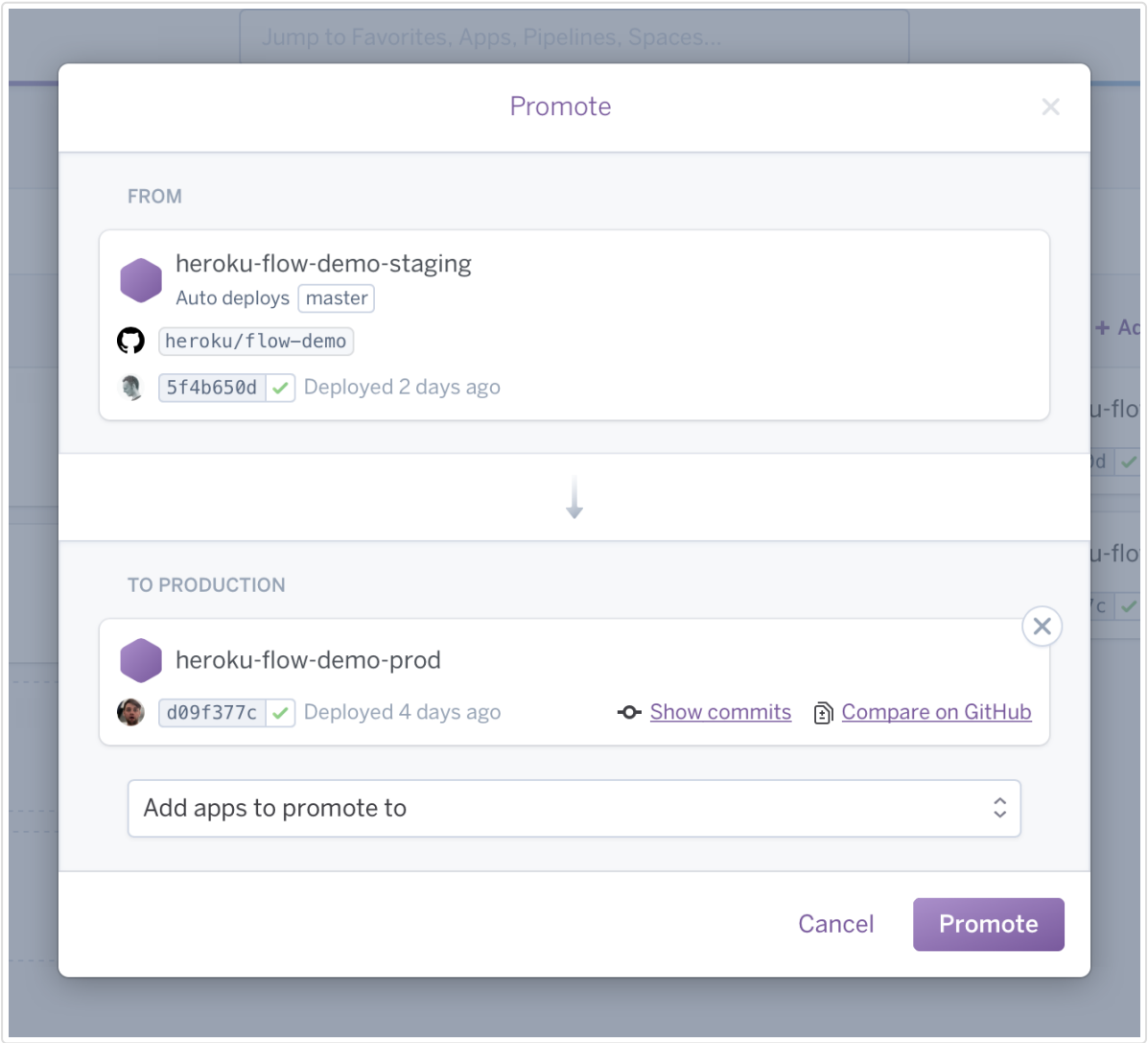
Promoting

After you've fully tested a change in a particular pipeline stage, you can promote its associated slug to the app(s) in the downstream stage.

> **Downstream** refers to the next environment stage in a pipeline. For example, given a `development --> staging --> production` pipeline, staging is downstream of development, and production is downstream of staging.

From the Heroku Dashboard

You promote a particular stage's slug by clicking its associated **`Promote`** button on your pipeline's overview page:



If there are multiple apps in the downstream stage, you can specify which ones you want to promote the slug to.

From the Heroku CLI

From the CLI, you can promote a slug with the `heroku pipelines:promote` command. The command must specify the name (`-a`) or Git remote (`-r`) of the source app:

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

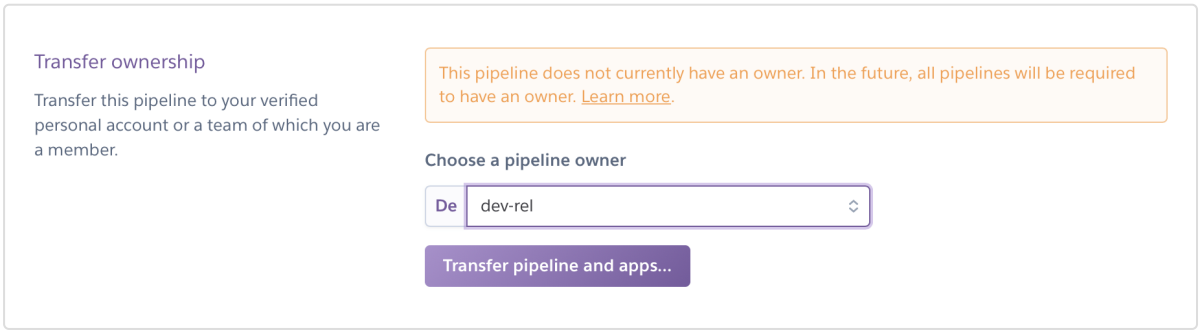
By default, this command promotes the source app's slug to *all* apps in the downstream stage. You can specify a subset of these apps to promote to with the `--to` option:

```
$ heroku pipelines:promote -r 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
```

Release phase during a promotion

Release Phase (https://devcenter.heroku.com/articles/release-phase) *does* run when you promote a slug, despite the fact that no new build is created.

Pipelines ownership and transfer



The Pipelines web interface and CLI encourage (and will eventually require (https://devcenter.heroku.com/articles/pipeline-ownership-transition)) that apps in a Pipeline be owned by the *Pipeline owner*. This owner can be assigned in the `Settings` tab of the Pipelines web interface.

Electing to assign a Pipeline owner, typically a Heroku Team, will prevent common permissions-related issues in collaborative workflows. This is especially important when owners wish to assign team members differing access to production, staging, and development apps.

This feature encourages better structure and administrative hierarchy in Pipelines.

A user is eligible to own a Pipeline if the user owns app(s) in a Pipeline. Once a user assumes ownership of the Pipeline, the web UI will highlight Pipeline apps not owned by that user and provide a UI to transfer the foreign apps in the Pipeline to the Pipeline owner.

Pipelines owned by an individual can only be transferred to a Heroku Team (or Enterprise Team) in which that individual are an member. Team-owned Pipelines can be transferred to any individual that is a member of that Team. However, for billing security, individual Pipelines cannot be transferred directly to other individuals.

GitHub Sync

Promoting from staging to production is great, but you can further optimize your flow by automatically deploying to staging using GitHub integration (https://devcenter.heroku.com/articles/github-integration). For example, whenever the `master` branch is updated on GitHub and continuous integration (CI) tests pass, staging can be auto-deployed.

Review apps

If you're using GitHub, we highly recommend using pull requests and setting up corresponding review apps (https://devcenter.heroku.com/articles/github-integration-review-apps).

Pipelines and Heroku CI

If you're using GitHub, you can turn on Heroku CI (https://devcenter.heroku.com/articles/heroku-ci), Heroku's visual, low-configuration test runner that integrates easily with Heroku Pipelines. Any Heroku Pipeline is already CI-ready – just turn it on in the pipeline's `Settings` tab.

Design considerations

Pipelines manage the flow of application slugs *only*. Your app's Git repo (https://devcenter.heroku.com/articles/git), config vars (https://devcenter.heroku.com/articles/config-vars), add-ons (https://devcenter.heroku.com/articles/managing-add-ons), and other environmental dependencies must be managed independently.

Deploying via pipeline promotion is recommended only for apps with **stateless builds**. Builds that compile configuration variable values into the slug (i.e., **stateful builds**) can encounter issues when promoted. For apps with stateful builds, use Heroku's standard Git-based deployment (https://devcenter.heroku.com/articles/git) or GitHub deploys (https://devcenter.heroku.com/articles/github-integration) instead.

When a slug is promoted, Heroku copies it without making any changes. It is *not* rebuilt for the environment of the target app. If your builds bake in environment from the build-app context, then your slugs are not portable between pipeline stages. This may be the case, for example, if you're using Ruby on Rails and the asset pipeline to build assets hosted at a CDN defined by a URL in your build environment. This is because URLs specific to the build-app will be baked into css and js-files, and those will not work correctly when promoted. Please see this article (https://devcenter.heroku.com/articles/please-do-not-use-asset-sync#pipelines) for instructions on how to configure this with Rails.

FAQ

What else can I do with the pipelines CLI?

A complete list of Pipelines commands with usage details is available in the console:

```
$ heroku help pipelines
```

The repository README for the Heroku CLI plugin (https://github.com/heroku/heroku-pipelines) provides additional usage examples and a feature history.

Can I have pipelines stages other than development, staging and production?

No, only those three stages are currently supported, plus the special stage, review.

Can I run scripts, such as rake db:migrate when promoting?

Heroku Release Phase (https://devcenter.heroku.com/articles/release-phase) lets you perform common tasks like schema migrations before a new version of your app is deployed to any step of the continuous delivery flow. See its documentation for more information.

Can I have more than one app in a pipeline stage?

Yes.

Do pipelines work without GitHub?

Yes. Pipelines are very well integrated with GitHub, but this integration is not required.

I don't see a "development" stage, how do I add a development app?

You can add the app to any other stage, and then using the context menu on the app card, move the app to “development”. Alternatively, you can go to the **`Deploy`** tab of the app and add it to a pipeline from there, while specifying the “development” stage.

Do Pipelines work with Heroku Enterprise Teams?

Yes, Pipelines are fully supported for Enterprise Teams. However, in some cases team members may find they cannot operate on a given app in a pipeline due to the access control, such as that described in Using App Permissions in Heroku Enterprise Teams (https://devcenter.heroku.com/articles/app-privileges-in-heroku-organizations). Users must have adequate permissions to perform the relevant operations on the app.

Do Pipelines work with Private Spaces?

Yes, a Pipeline can span apps in the Common Runtime and in multiple different Private Spaces. This allows you to have test and staging apps in the Common Runtime or in a separate Private Space while promoting to a Private Space with only production apps.