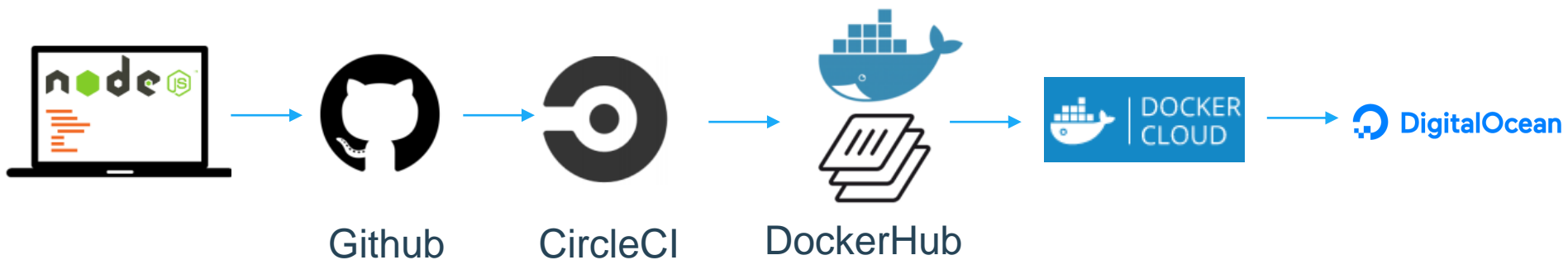


- Linas- 159.65.126.44
- Tadas - 159.65.114.171
- Vaidotas - 159.65.114.45
- Aurelijus - 159.65.126.115
- Lolita – 159.65.124.113
- Mindaugas – 159.89.99.102
- Deimantas - 159.65.118.52
- Rimantas – 159.65.118.204
- Nerijus - 159.65.118.189
- Ignas - 159.65.126.175

Docker in Continuous Integration



CI/CD workflow



App

<https://github.com/docker-4-devops/docker-ci>

- Simple JS app
- Dockerfile to build image
- Mocha to test the App

Express

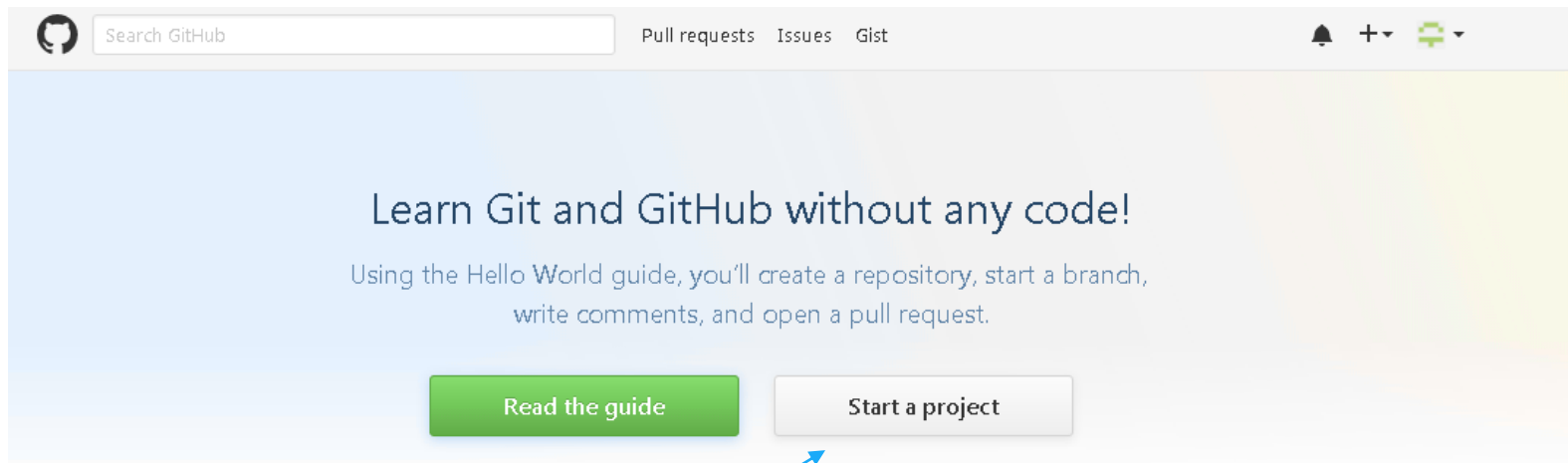


Webhooks & Triggers

- Event-based triggers
- Crucial part of our CI/CD Workflow



Start new project on github



Create repository

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name



docker-4-devops ▾

/

docker-ci



Great repository names are short and memorable. Need inspiration? How about **solid-chainsaw**.

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾



Create repository

Import code

Quick setup — if you've done this kind of thing before

 Set up in Desktop or **HTTPS** **SSH** <https://github.com/jomajo/docker-ci.git>

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

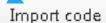
```
echo "# docker-ci" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/jomajo/docker-ci.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/jomajo/docker-ci.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.



Import existing project

<https://github.com/docker-4-devops/docker-ci>

Import your project to GitHub

Import all the files, including the revision history, from another version control system.

Your old repository's clone URL

Learn more about the types of [supported VCS](#).

Your existing repository

 docker-4-devops/**docker-ci2**

[Change repository](#)

[Cancel](#)

[Begin import](#)

Sign up at CircleCi

<https://circleci.com/signup/>



Step1

You're almost there.
Now let's add some code.

Signing up with CircleCI is **free**. Next, you'll be taken to GitHub or Bitbucket to authenticate so you can start shipping faster.

Authorize GitHub

Authorize Bitbucket

By clicking on "Authorize GitHub" or "Authorize Bitbucket" you are agreeing to our [Terms of Service](#) and [Privacy Policy](#).



Begin **signup**



Add **code**



Start **testing**

Step2

Review permissions



Personal user data

Email addresses (read-only)



Repositories

Public and private



Authorize application

Add project to Circle Ci

Add Projects Reload Organizations

1 Choose an organization that you are a member of.

2 Choose a repo to add to CircleCI. We'll start a new build for you each time someone pushes a new commit. You can also follow a repo that's already been added to CircleCI. You'll see your followed projects in [Builds](#) and [Insights](#).

GitHub **Bitbucket**

Are you missing an organization? You or an admin may need to enable CircleCI for your organization in [GitHub's application permissions](#). [Refresh this list](#) after you have updated permissions.

docker-4-devops

Linux **OSX**

Filter repos...

☒ Show Forks

docker-ci2 Build project

docker-ci Build project

This project is not building on CircleCI. Clicking will cause CircleCI to start building the proj

Add project to Circle Ci

The screenshot shows the CircleCI interface for setting up a new project. The top navigation bar includes the CircleCI logo, the project name 'docker-4-devops', and links for 'Updates' and 'Support'. A left sidebar contains navigation icons for 'BUILDS', 'WORKFLOWS', 'INSIGHTS', 'ADD PROJECTS', 'TEAM', and 'SETTINGS'. The main content area is titled 'Projects » Add Projects » docker-4-devops/docker-ci'.

Set Up Project

CircleCI helps you ship better code, faster. To kick things off, you'll need to add a `config.yml` file to your project, and start building. After that, we'll start a new build for you each time someone pushes a new commit. Select from the following options to generate a sample `.yml` for your project.

Operating System

Linux macOS

Language

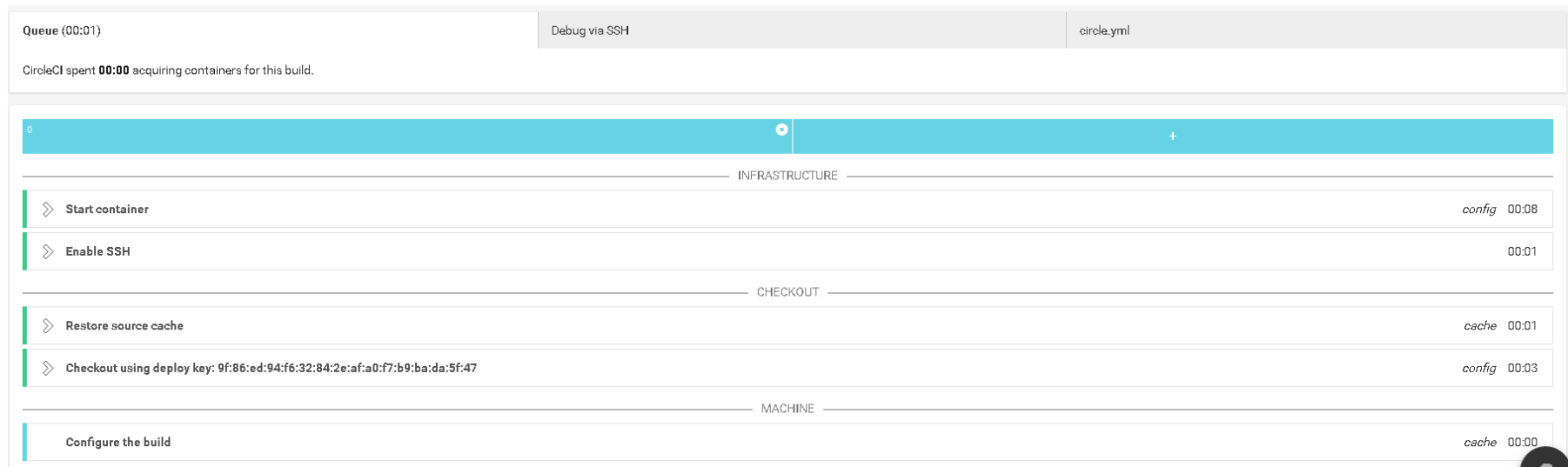
Clojure Elixir Go Gradle (Java) Maven (Java) Node PHP Python Ruby Scala Other

Next Steps

You're almost there! We're going to walk you through setting up a configuration file, committing it, and turning on our listener so that CircleCI can test your commits. Want to skip ahead? Jump right [into our documentation](#), set up a `.yml` file, and kick off your build with the button below.

- Create a folder named `.circleci` and add a file `config.yml` (so that the filepath be in `.circleci/config.yml`).
- Populate the `config.yml` with the contents of the sample `.yml` (shown below). [Copy to clipboard](#)
- Update the sample `.yml` to reflect your project's configuration.
- Push this change up to GitHub.
- Start building! This will launch your project on CircleCI and make our webhooks listen for updates to your work. [Start building](#)

Build starts immediately!



Check if tests are passing!

♡ \$ mocha

```
$ mocha
```

```
GET /
```

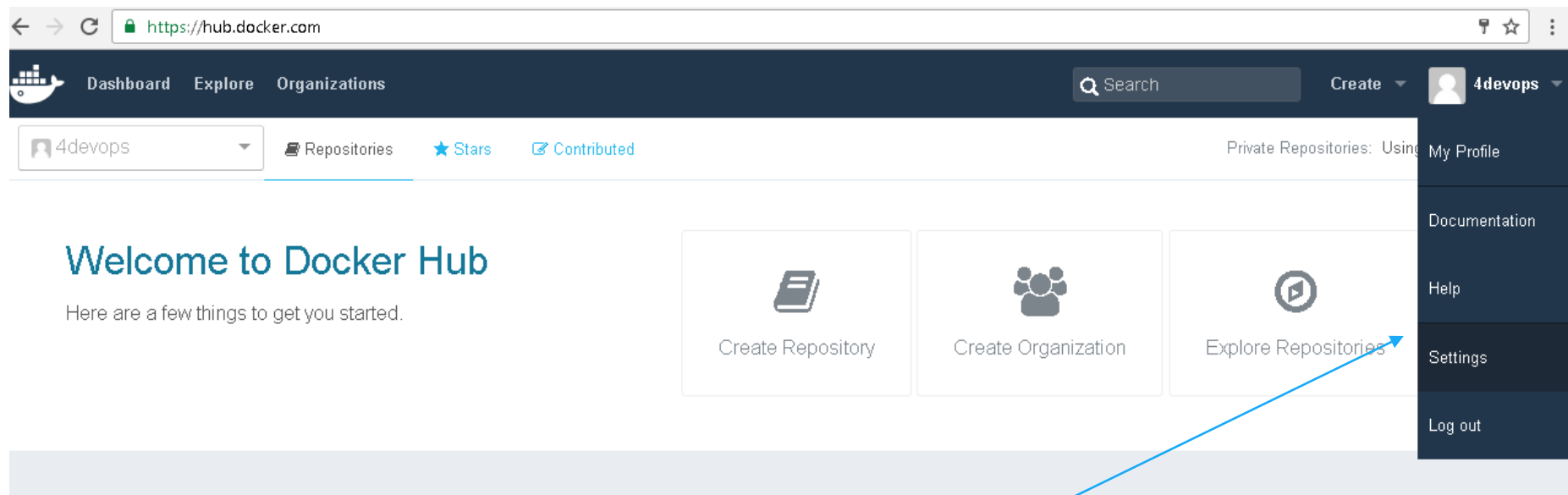
```
✓ expects HTTP response 200 (342ms)
```

```
1 passing (351ms)
```

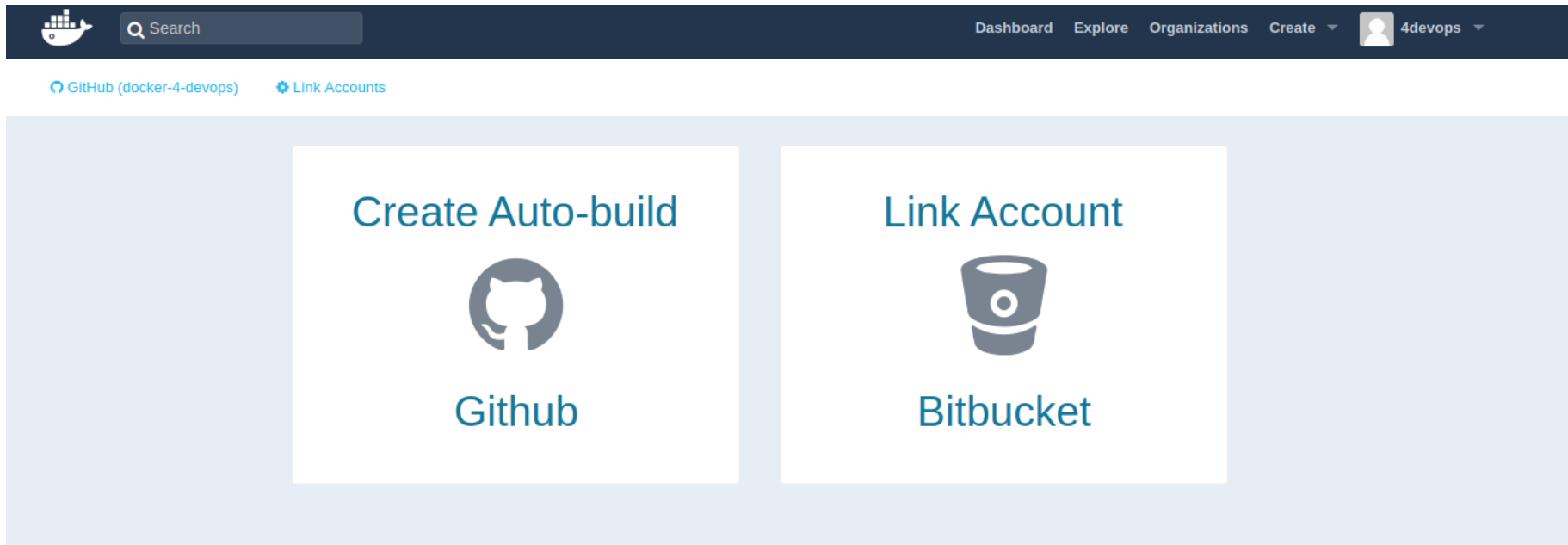
Next-step - Dockerhub

<https://hub.docker.com/>

- Link Github account by going to:
 - Settings -> Linked Accounts & Services



Link Your github account



Link Github account to DockerHub

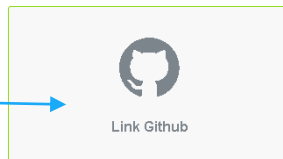


Linked Accounts & Services

Step1

Linked Accounts

These account links are currently used for Automated Builds, so that we can access your project lists and help you configure your Automated Builds. **Please note: A github/bucket account can be connected to only one docker hub account at a time.**



Step2

Public and Private (Recommended)

- Read and Write access to public and private repositories. (We only use write access to add service hooks and add deploy keys)
- Required if you want to setup an Automated Build from a private GitHub repository.
- Required if you want to use a private GitHub organization.
- We will automatically configure the service hooks and deploy keys for you.

Select

Authorize application

Docker Hub Registry by @docker would like permission to access your account

Step3

Review permissions



Repositories

Public and private

Authorize application

Step4

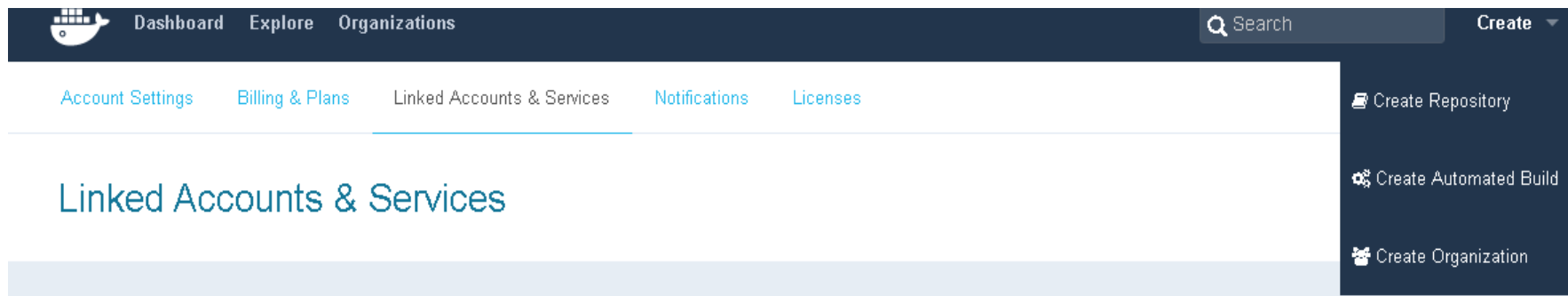


docker-4-devops:
read/write access

Unlink Github

Create automated build repository on DockerHub

- Create automated build repository



Create automated build repository on DockerHub

Create Automated Build

Repository Namespace & Name*

4devops ▾

docker-ci

Visibility

public ▾

Short Description*

Max 100 Characters

By default Automated Builds will match branch names to Docker build tags. [Click here to customize](#) behavior.

Create

Automated Build Repository

1devops/docker-ci ☆

ast pushed: never

Repo Info

Tags

Dockerfile

Build Details

Build Settings

Collaborators

Webhooks

Settings

Short Description



docker-ci

Full Description



Full description is empty for this repo.

Uncheck the option below under “Build Settings”

We want to build image only if tests are passing!

Build Settings

☐ When active, builds will happen automatically on pushes.

The build rules below specify how to build your source into Docker images. The name can be a string or a regex. The Docker Tag name may contain variables. We currently support {sourcerefer}, which refers to the source branch/tag name. [Show more](#)

Activate Triggers under “Build Settings”

Build Triggers

Trigger your Automated Build by sending a POST to a specific endpoint.

Activate Triggers

Step1

Trigger endpoints are activated. Use the trigger token or URL below in your requests. [Show examples.](#)

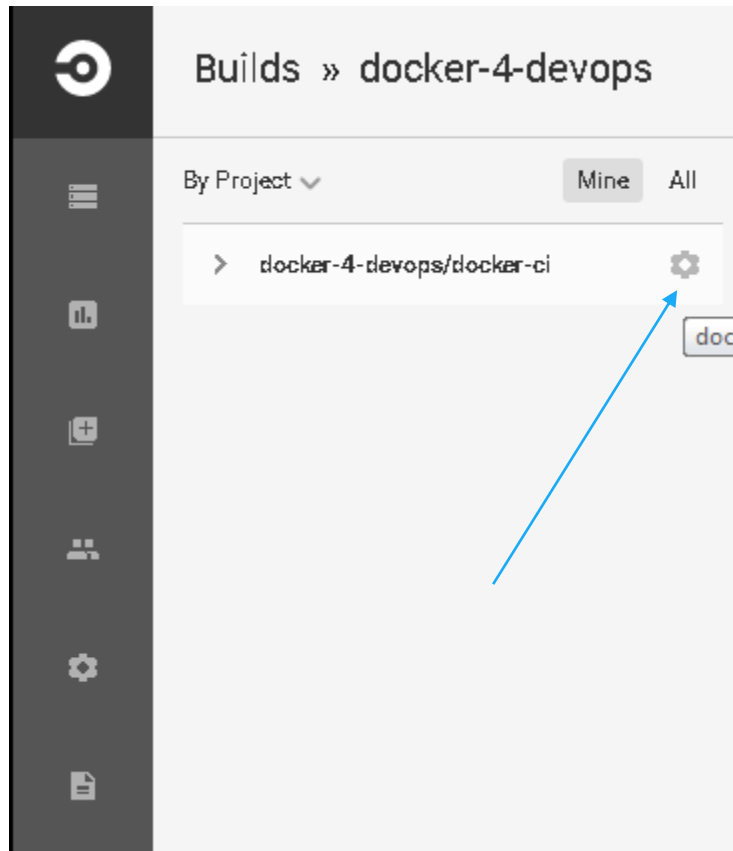
Step2

Examples

```
# Trigger all tags/branches for this automated build.  
$ curl -H "Content-Type: application/json" --data '{"build": true}' -X POST https://registry.hub.docker.com/u/4devops/docki
```

Step3

Copy Trigger URL to Circle CI



Add Trigger as Environment Variable

Under Build Settings -> Environmental Variables

- Name:

- Value:

Add an Environment Variable

×

To disable string substitution you need to escape the `$` characters by prefixing them with `\`. For example, a value like `usd$` would be entered as `usd\$`.

pository)

Trigger URL

`https://registry.`

Name

`DOCKER_HUB_TRIGGER`

Value

`curl -H "Content-Type: application/json" -data '{"build": true}' -X POST https://registry.hub.docker.com/u/4`

Settings » evaldasou

PROJECT SETTINGS

[Overview](#)

[Org Settings](#)

BUILD SETTINGS

[Build Environment](#)

[Adjust Parallelism](#)

[Environment Variables](#)

[View docker-ci](#)

Add Variable

Cancel

Add Variable

3, such as setting `M2_MAVEN` to

Remove

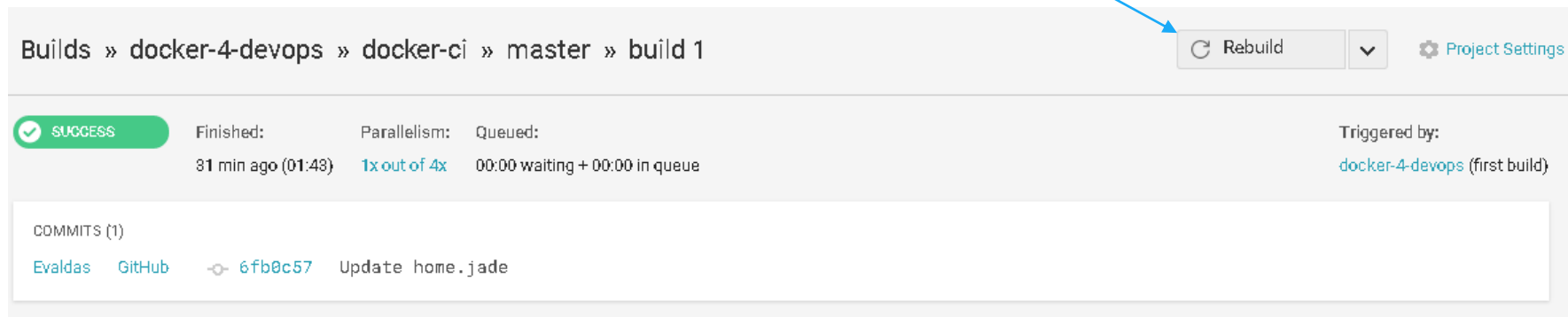
`DOCKER_HUB_TRIGGER`

`xxxx86c/`

×

Rebuild code under CircleCi

To Trigger Dockerfile build on DockerHub



The screenshot shows the CircleCI interface for a build. At the top, the breadcrumb navigation reads: Builds » docker-4-devops » docker-ci » master » build 1. On the right side of this bar, there is a 'Rebuild' button with a circular arrow icon, a dropdown menu, and a 'Project Settings' link with a gear icon. A blue arrow points from the top right towards the 'Rebuild' button. Below the breadcrumb bar, the build status is 'SUCCESS', indicated by a green checkmark and the word 'SUCCESS' in a green box. To the right of the status, build details are shown: 'Finished: 31 min ago (01:43)', 'Parallelism: 1x out of 4x', and 'Queued: 00:00 waiting + 00:00 in queue'. Further right, it says 'Triggered by: docker-4-devops (first build)'. Below this, a section titled 'COMMIT (1)' shows a commit from 'Evaldas' on 'GitHub' with the hash '6fb0c57' and the message 'Update home.jade'.

Builds » docker-4-devops » docker-ci » master » build 1

Rebuild

Project Settings

SUCCESS

Finished: 31 min ago (01:43) Parallelism: 1x out of 4x Queued: 00:00 waiting + 00:00 in queue

Triggered by: docker-4-devops (first build)

COMMIT (1)

Evaldas GitHub 6fb0c57 Update home.jade

Do we have new docker image on DockerHub?

- Check if app tests have passed on Circle CI
- Image was build and saved to Dockerhub
(image build from Dockerfile (which is stored on Github))

4devops/docker-ci ☆

_last pushed: an hour ago

[Repo Info](#)

[Tags](#)

[Dockerfile](#)

[Build Details](#)

[Build Settings](#)

[Collaborators](#)

[Webhooks](#)

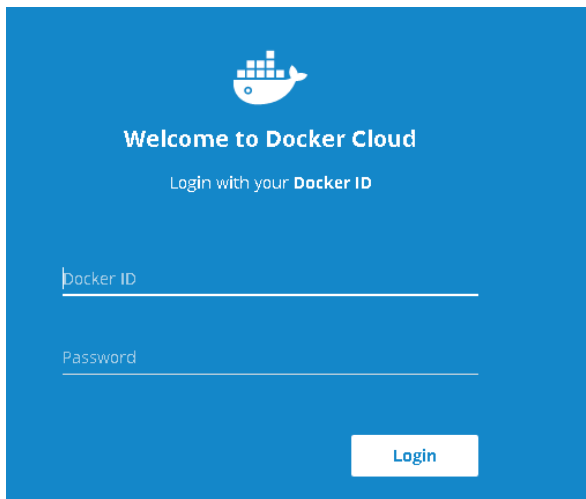
[Settings](#)

Status	Actions	Tag	Created	Last Updated
✓ Success		latest	an hour ago	an hour ago

Login to DockerCloud

<https://cloud.docker.com/>

- For DockerCloud you can use same account as for Dockerhub
 - Choose login with Docker ID option.

The image shows a login interface for Docker Cloud. It features a blue background with a white Docker logo (a ship) at the top center. Below the logo, the text "Welcome to Docker Cloud" is displayed in white. Underneath, it says "Login with your Docker ID". There are two input fields: one labeled "Docker ID" and another labeled "Password". Both fields have white text on a blue background. At the bottom right, there is a white button with the text "Login" in blue.

Docker Cloud

The screenshot displays the Docker Cloud web interface. On the left is a sidebar with navigation links: Swarm Mode (with a BETA badge), BUILD (Repositories), APPLICATIONS (Stacks, Services, Containers), INFRASTRUCTURE (Node Clusters, Nodes), SETTINGS (Cloud Settings), and Status. The main content area has a blue header with the Docker Cloud logo, a plus icon, 'Get Help', and a user profile '4devops'. Below the header is a 'Welcome!' banner. The main heading is 'Welcome to Docker Cloud!' with a subtext 'Let's get you familiarized with the central concepts of Docker Cloud.' A horizontal menu contains 'Cloud registry' (selected), 'Continuous integration', 'Application deployment', 'Continuous deployment', and 'Teams & Organizations'. The 'Cloud registry' section is titled 'Cloud registry' and includes the text 'Create and share private image repositories securely with your teams, or make them public to share them with the entire community.' Below this is the question 'When should I use the Cloud Registry?' followed by two links: 'To create public or private image repositories' and 'To set up an Automated Build for repositories'. At the bottom is a blue button labeled 'Create repository +'.

←

Swarm Mode **BETA**

BUILD

Repositories

APPLICATIONS

Stacks

Services

Containers

INFRASTRUCTURE

Node Clusters

Nodes

SETTINGS

Cloud Settings

Status

docker cloud + Get Help 4devops

Welcome!

Welcome to Docker Cloud!

Let's get you familiarized with the central concepts of Docker Cloud.

[Cloud registry](#) Continuous integration Application deployment Continuous deployment Teams & Organizations

Cloud registry

Create and share private image repositories securely with your teams, or make them public to share them with the entire community.

When should I use the Cloud Registry?

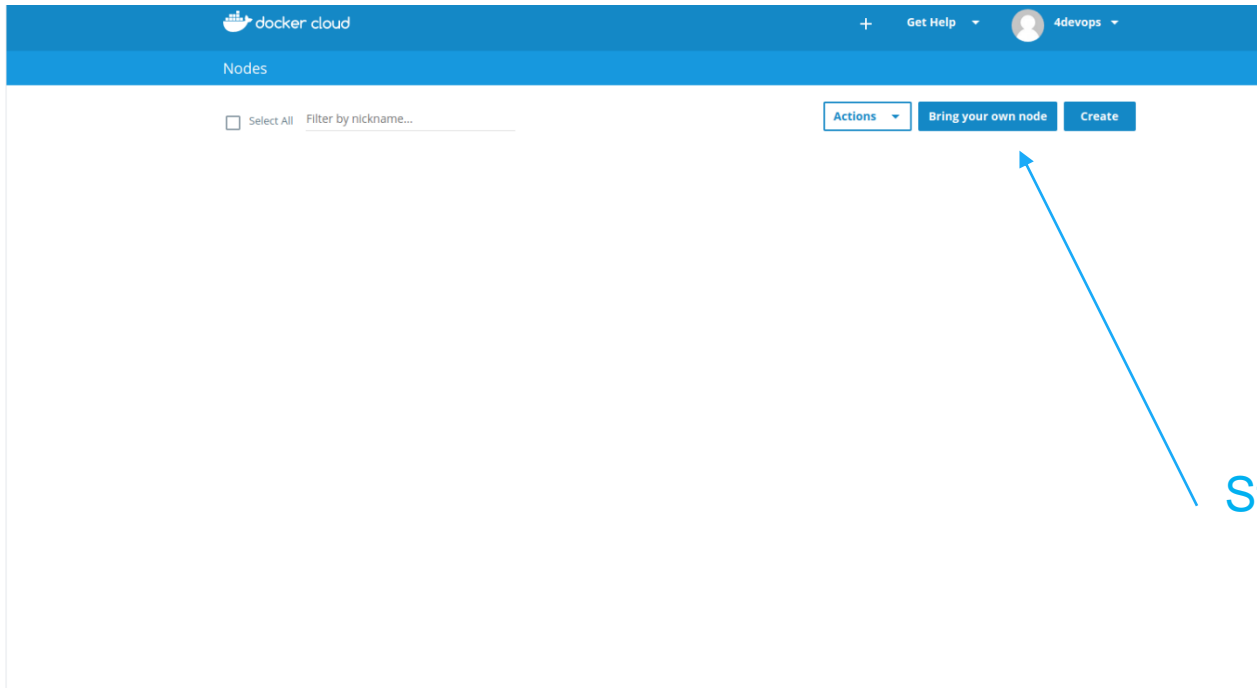
[To create public or private image repositories](#)

[To set up an Automated Build for repositories](#)

Create repository +

Create a node on Docker Cloud

Step1



Step2

Bring your own Node

Bring your own Node

Docker Cloud lets you use your own host as a node to run containers. In order to do this, you have to first install the Docker Cloud Agent.

The following Linux distributions are supported:



Ubuntu
14.04, 15.04



Debian 8



Centos 7



RedHat
Linux 7



Fedora
21, 22

Run the following command in your Linux host to install the Docker Cloud Agent or click [here](#) to learn more:

```
curl -Ls https://get.cloud.docker.com/ | sudo -H sh -s 25db41f606714f22a2dcfba55b4aad9a
```

We recommend you open incoming port 2375 in your firewall for Docker Cloud to communicate with the Docker daemon running in the node. For the overlay network to work, you must open port 6783/tcp and 6783/udp.

· Waiting for contact from agent

Let me spin up Digital Ocean nodes for You

- `docker run -it 4devops/ssh`
- `ssh IP-ADDRESS`
- For example: `ssh 104.2.2.2`
- `curl -Ls https://get.cloud.docker.com/ | sudo -H sh -s 0b35c36f027148b98f34fde97769b523`



- Linas- 159.65.126.44
- Tadas - 159.65.114.171
- Vaidotas - 159.65.114.45
- Aurelijus - 159.65.126.115
- Lolita – 159.65.124.113
- Mindaugas – 159.89.99.102
- Deimantas - 159.65.118.52
- Rimantas – 159.65.118.204
- Nerijus - 159.65.118.189
- Ignas - 159.65.126.175

Deploying Node

←

BETA

Swarm Mode

BUILD

Repositories

APPLICATIONS

Stacks

Services

Containers

INFRASTRUCTURE

Node Clusters

Nodes

SETTINGS

Cloud Settings

docker cloud

+

Get Help

4devops

Nodes

General

Containers

Timeline

91bac6ac-b9c1-422e-9449-f1c...

DEPLOYING

IP 159.65.114.67

CLOUD PROVIDER

REGION -

NODE TYPE BYON

MEMORY -

DISK SPACE -

DOCKER INFO

Version 1.11.2-cs5

Graph driver

Exec driver

Endpoints

No endpoint defined

Terminate

33

Deployed Node

←

Swarm Mode

BUILD

REPOSITORIES

APPLICATIONS

Stacks

Services

Containers

INFRASTRUCTURE

Node Clusters

Nodes

SETTINGS

Cloud Settings

Status

docker cloud

+

Get Help

4devops

Nodes

GeneralContainersTimeline

Terminate

ad271f62-143e-4b5d-9488-1f7...

DEPLOYED

a few seconds ago

IP 207.154.249.52

CLOUD PROVIDER

REGION -

NODE TYPE BYON

MEMORY -

DISK SPACE -

DOCKER INFO

Version 1.11.2-cs5

Graph driver aufs

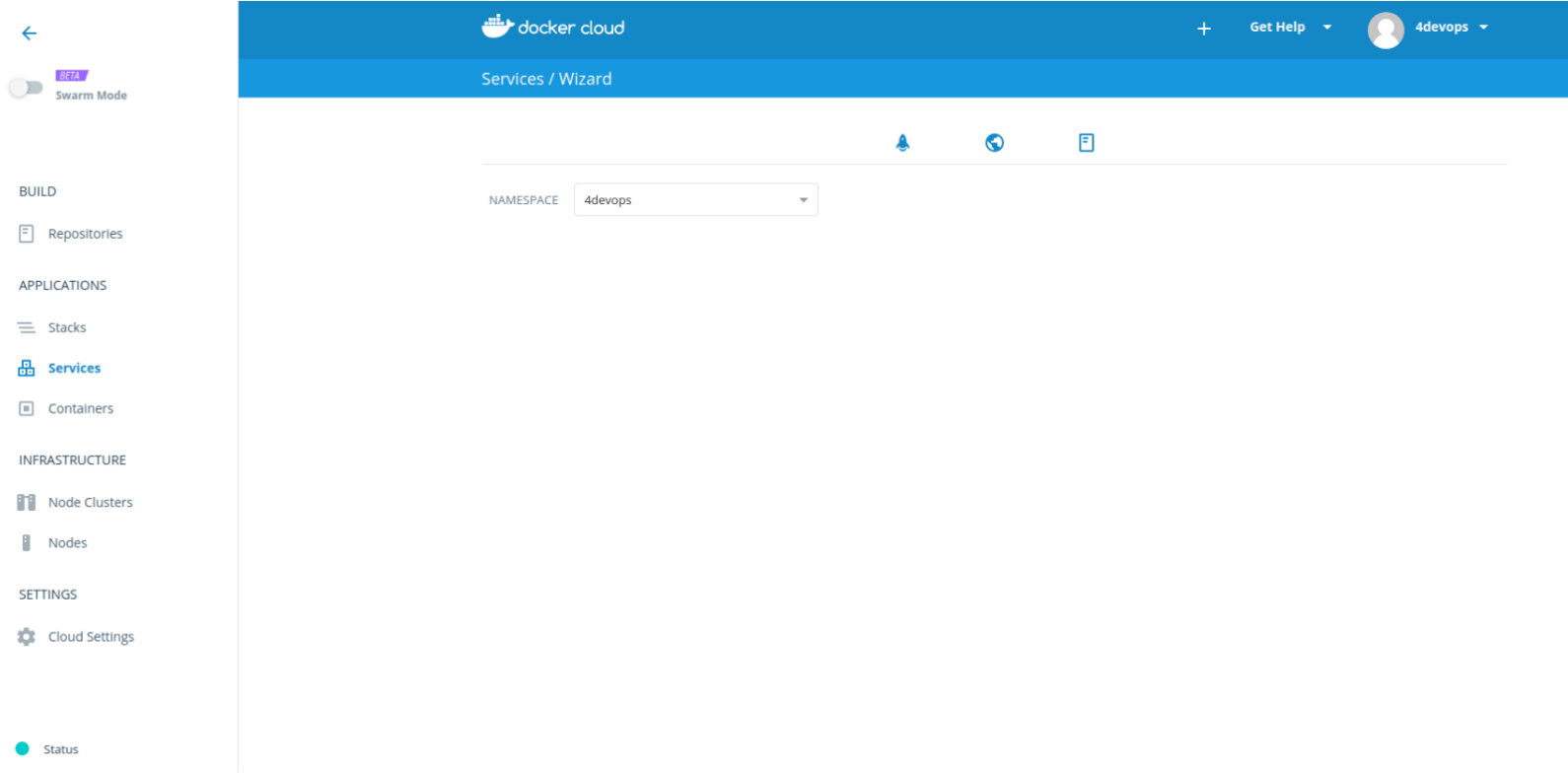
Exec driver

Endpoints

No endpoint defined

34

Create new service



Select repository



4devops/docker-ci

docker-ci

🕒 3 hours ago

Select




4devops/ssh

🕒 16 minutes ago

Expose port when creating service

Ports

Container port	Protocol	Published	Node port	
8080	tcp ▾	<input checked="" type="checkbox"/>	80 ▴ ▾	

Use image values

Add Port

Create & Deploy

DEPLOYMENT STRATEGY	<div>Emptiest Node ▼</div>
DEPLOYMENT CONSTRAINTS	<div>Select... ▼</div>
AUTORESTART	<div><input checked="" type="radio"/> Off <input type="radio"/> On failure <input type="radio"/> Always</div>
AUTODESTROY	<div><input checked="" type="radio"/> Off <input type="radio"/> On success <input type="radio"/> Always</div>
SEQUENTIAL DEPLOYMENT	<div><input type="checkbox"/></div>
AUTOREDEPLOY	<div><input checked="" type="checkbox"/></div>
NETWORK	<div>bridge ▼</div>
PID	<div>none ▼</div>
API ROLES	<div>Select... ▼</div>

Trigger! To redeploy!

Create & Deploy

The screenshot shows the Docker Cloud 'Create & Deploy' wizard. The interface is divided into a left sidebar, a main configuration area, and a right summary panel.

Left Sidebar:

- Navigation icons for back, home, and user profile.
- Buttons for 'Swarm Mode' (disabled) and 'Status'.
- Menu categories: BUILD, APPLICATIONS, INFRASTRUCTURE, and SETTINGS.
- Sub-items: Repositories, Stacks, Services (selected), Containers, Node Clusters, Nodes, Cloud Settings.

Main Configuration Area (General settings):


- IMAGE:** 4devops/docker-ci, latest
- SERVICE NAME:** docker-ci-1f33e95a
- NICKNAME:** The alias of your service
- ADD TO STACK:** Select a Stack
- CONTAINERS:** 1
- DEPLOYMENT STRATEGY:** Emptiest Node
- DEPLOYMENT CONSTRAINTS:** Select...
- AUTORESTART:** Off (selected), On failure, Always
- AUTODESTROY:** Off (selected), On success, Always
- SEQUENTIAL DEPLOYMENT:** Toggle off
- AUTOREDEPLOY:** Toggle off
- NETWORK:** bridge

Right Summary Panel:

- SUMMARY**
- General settings
- Container configuration
- Ports
- Links
- Environment variables
- Volumes
- Create & Deploy** button with a dropdown arrow


A blue arrow points from the text 'Create & Deploy!' to the 'Create & Deploy' button in the summary panel.


Start service





DOCKER-CI-489F22E8


RUNNING


 a few seconds ago


 STACK NAME -


 IMAGE TAG 4devops/docker-ci:latest


 RUN COMMAND


 SEQUENTIAL DEPLOYMENT OFF


 DEPLOYMENT STRATEGY EMPTIEST_NODE


 PRIVILEGED MODE OFF

 AUTORESTART OFF

 AUTOREDEPLOY OFF

 AUTODESTROY OFF

 NETWORK bridge

 PORTS 8080→80/tcp

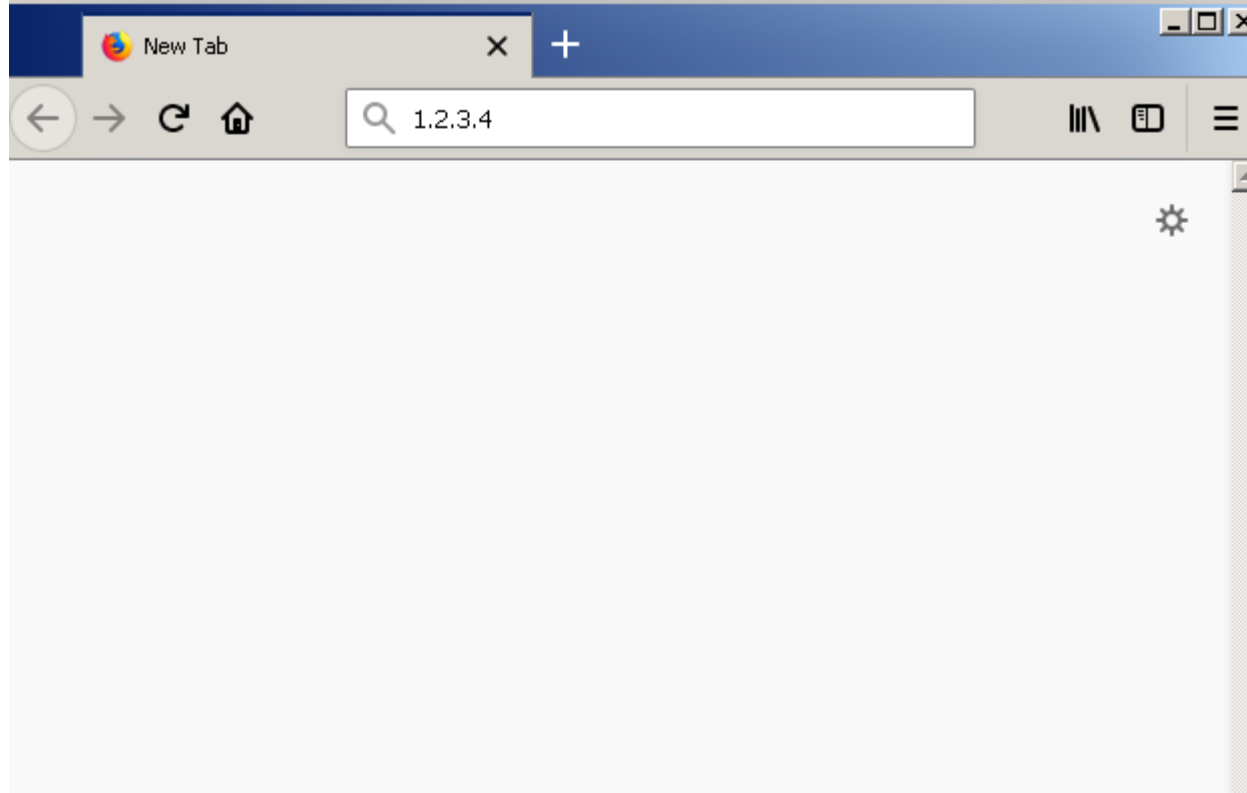
Edit

Actions ▾

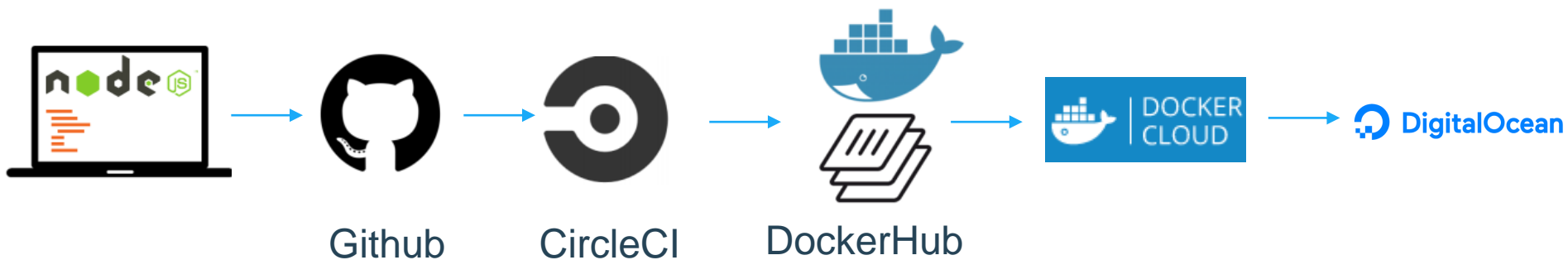
Stop

Check if Your service is reachable

Use IP address Your node has



Let's test our CI/CD workflow



Re-push code to DockerHub!

PUBLIC | AUTOMATED BUILD

4devops/docker-ci ☆

Last pushed: a minute ago


[Repo Info](#) [Tags](#) [Dockerfile](#) [Build Details](#) [Build Settings](#) [Collaborators](#) [Webhooks](#) [Settings](#)



Status	Actions	Tag	Created	Last Updated
✓ Success		latest	6 minutes ago	a few seconds ago



Source Repository

 [docker-4-devops/docker-ci](#)

Docker Cloud will redeploy app automatically!


 docker cloud


 Get Help 

 4devops 

Services


☐ Select All


Actions  Create





 1

docker-ci-1f33e95a

RUNNING


 docker-ci:latest

 3 minutes ago


1

Docker Cloud will redeploy app automatically!

 docker cloud

+





Get Help ▾

 4devops ▾

Containers

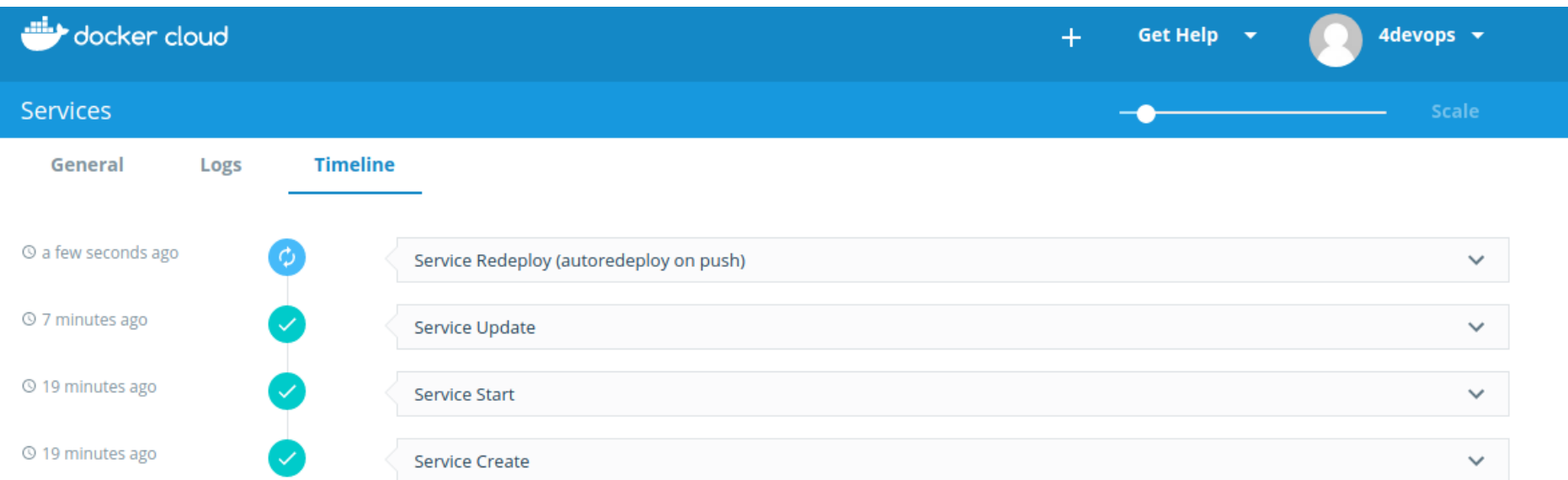
☐ Select All

Actions ▾

<input checked="" type="checkbox"/> docker-ci-1f33e95a-1	 docker-ci-1f33e95a	 docker-ci:latest	🕒 23 minutes ago
<input checked="" type="checkbox"/> docker-ci-1f33e95a-1	 docker-ci-1f33e95a	 docker-ci:latest	🕒 4 minutes ago

1

Timeline should show our continuous deployment



The screenshot displays the Docker Cloud interface. At the top, the 'docker cloud' logo is on the left, and a navigation bar on the right includes a plus sign, 'Get Help' with a dropdown arrow, a user profile icon, and '4devops' with a dropdown arrow. Below this is a 'Services' header with a slider and a 'Scale' button. The main content area has three tabs: 'General', 'Logs', and 'Timeline', with 'Timeline' being the active tab. The timeline shows four events:

Time	Status	Event
a few seconds ago	Refresh icon	Service Redeploy (autoredeploy on push)
7 minutes ago	Checkmark	Service Update
19 minutes ago	Checkmark	Service Start
19 minutes ago	Checkmark	Service Create



docker