

# dockerhubとgithubの連携

2020/12/18

木南 貴志

# はじめに

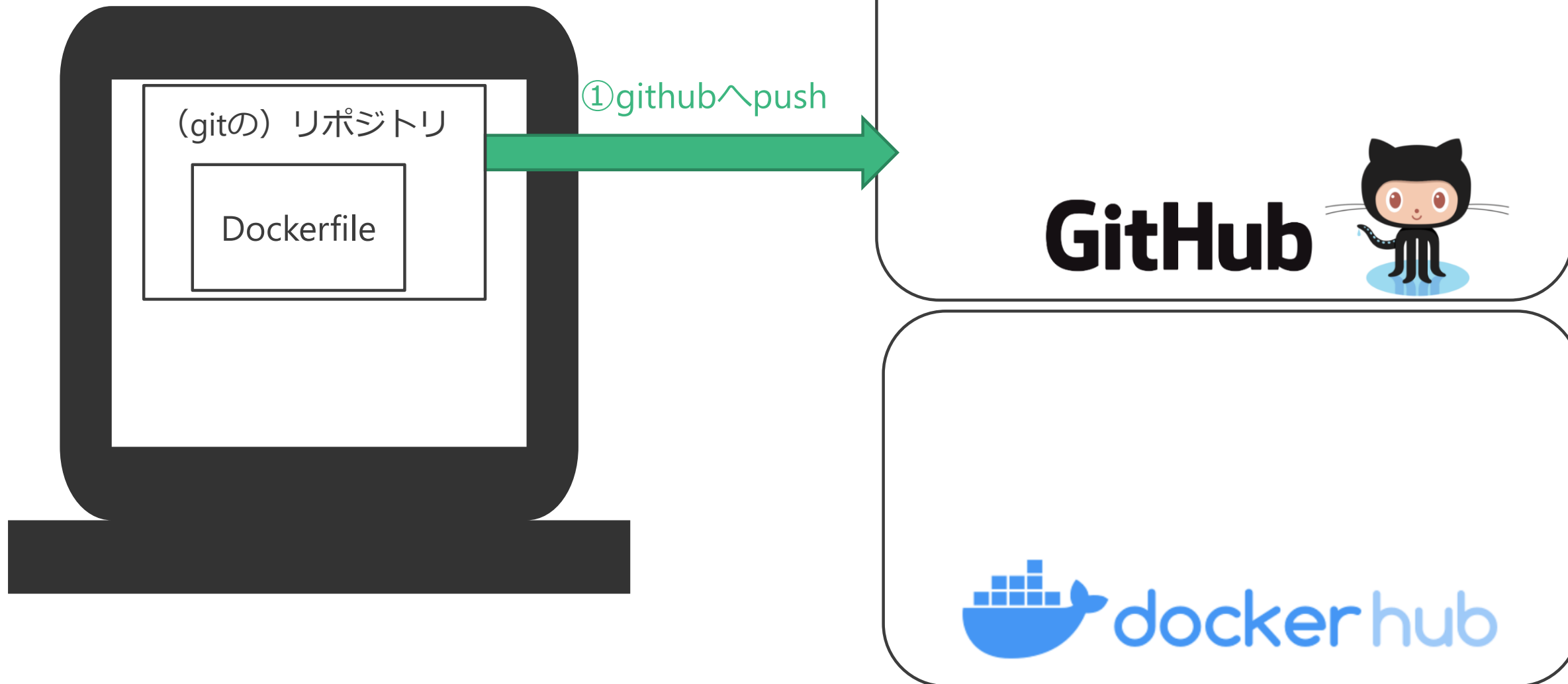
---

この資料は、「機械学習のためのdocker.pdf」でdocker関係のパッケージのインストール・設定，機械学習用のdocker環境の構築方法が理解できている前提で進めます。

また，dockerhubのアカウントの作成も完了している前提とします

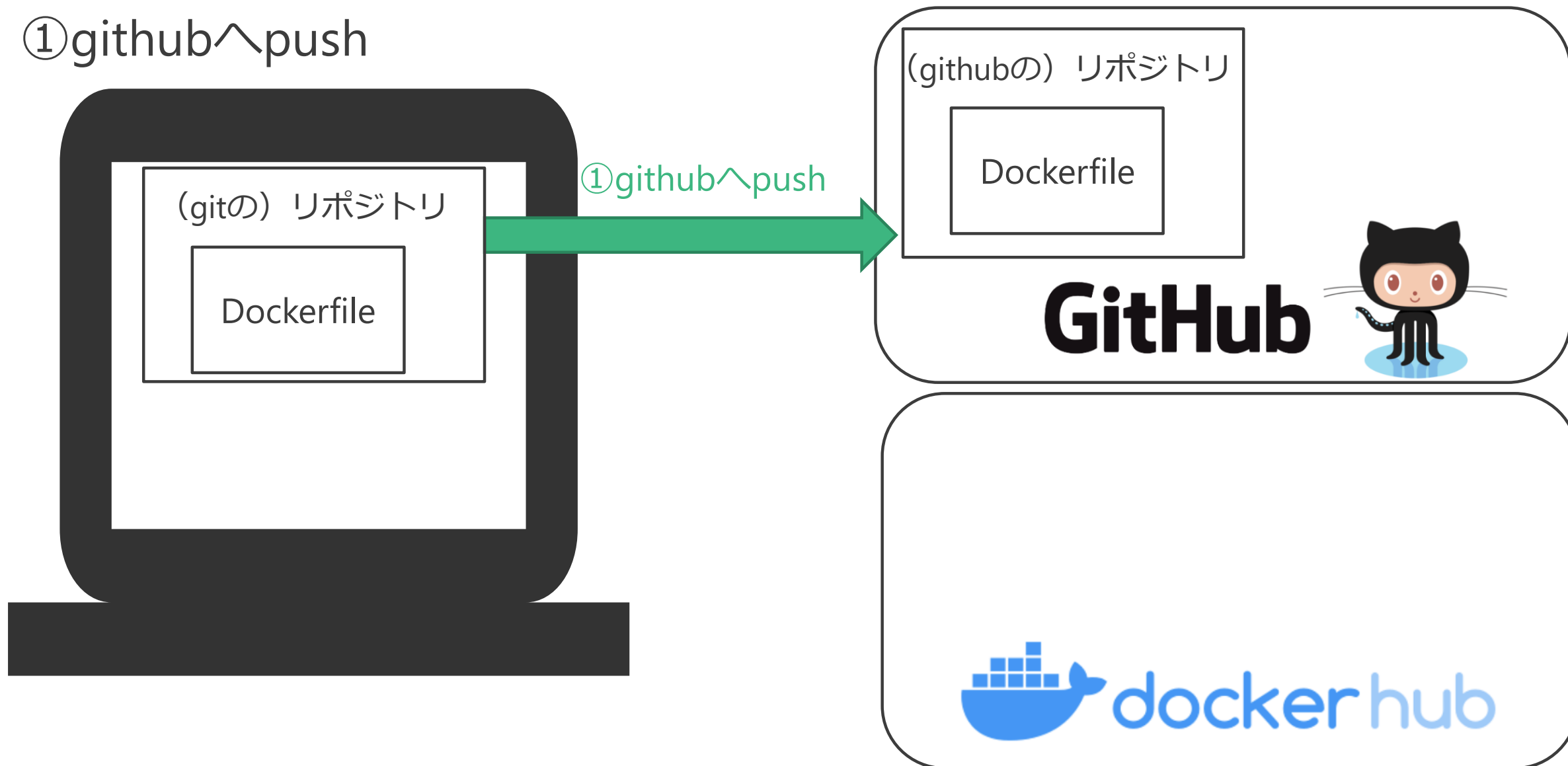
# *dockerhubとgithubの連携*

## ①githubへpush



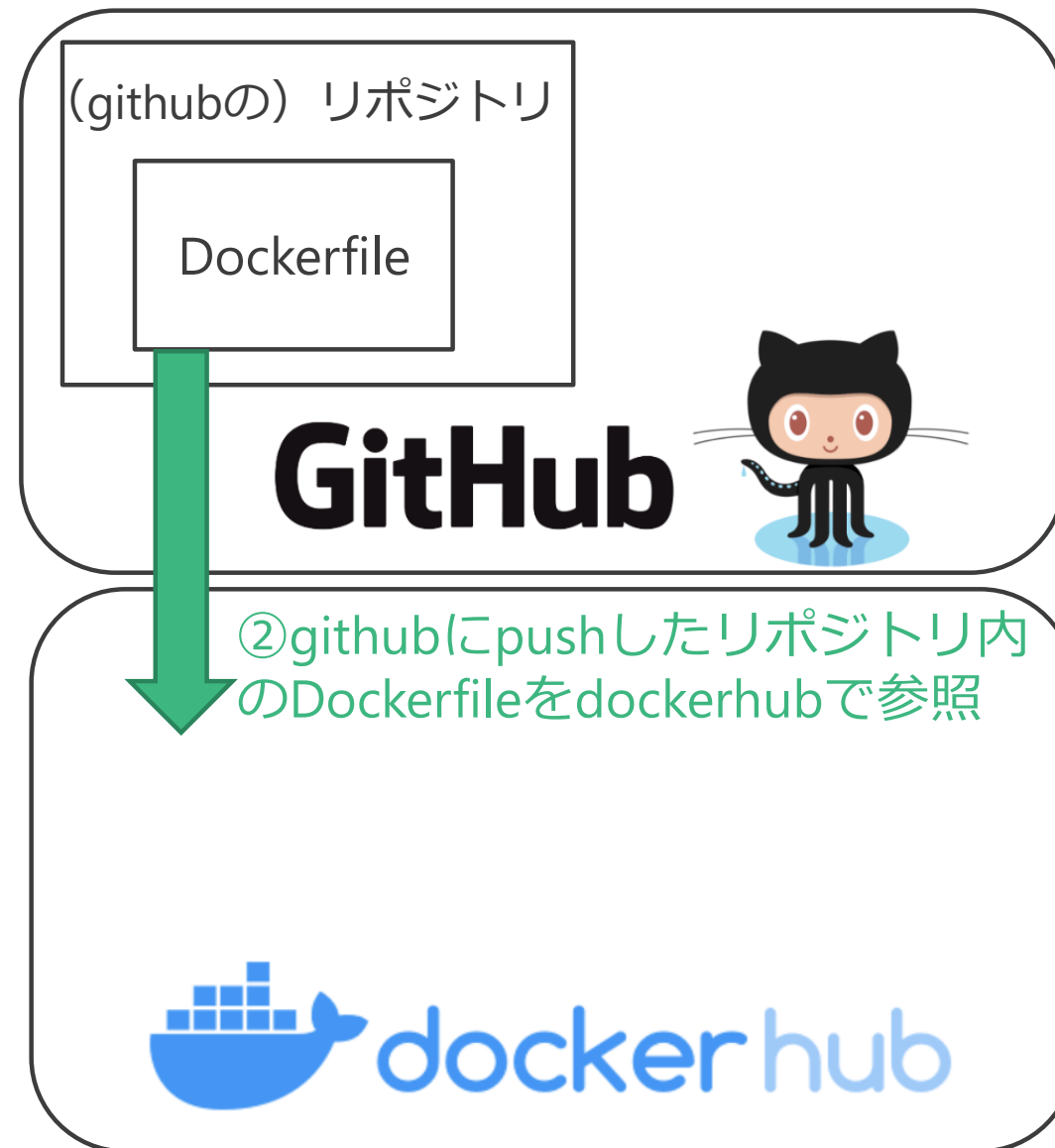
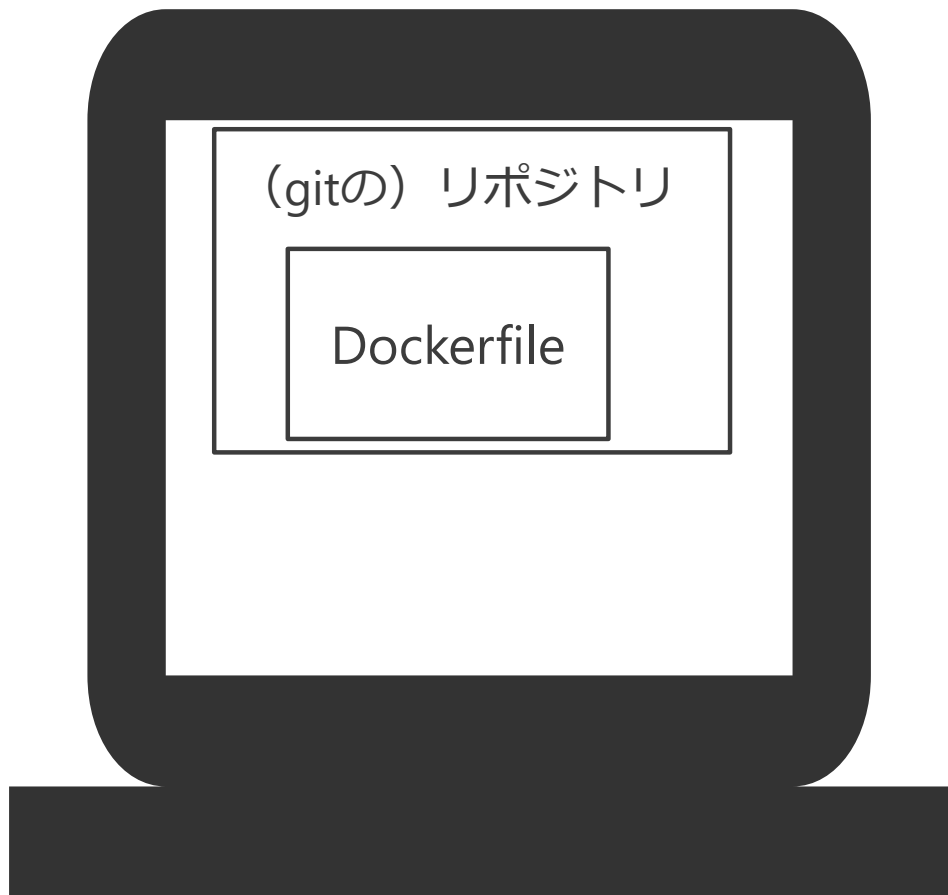
# *dockerhubとgithubの連携*

## ①githubへpush



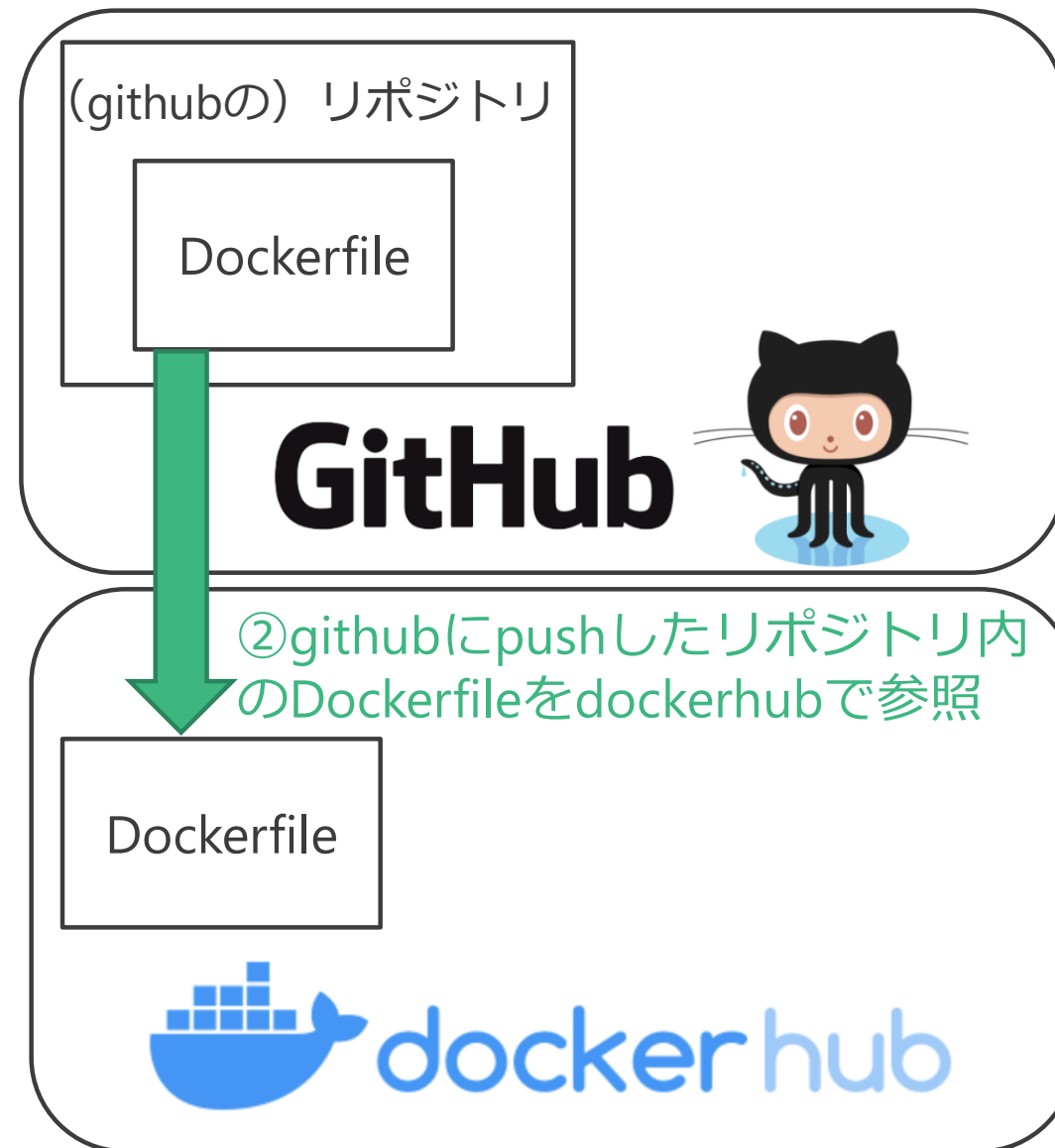
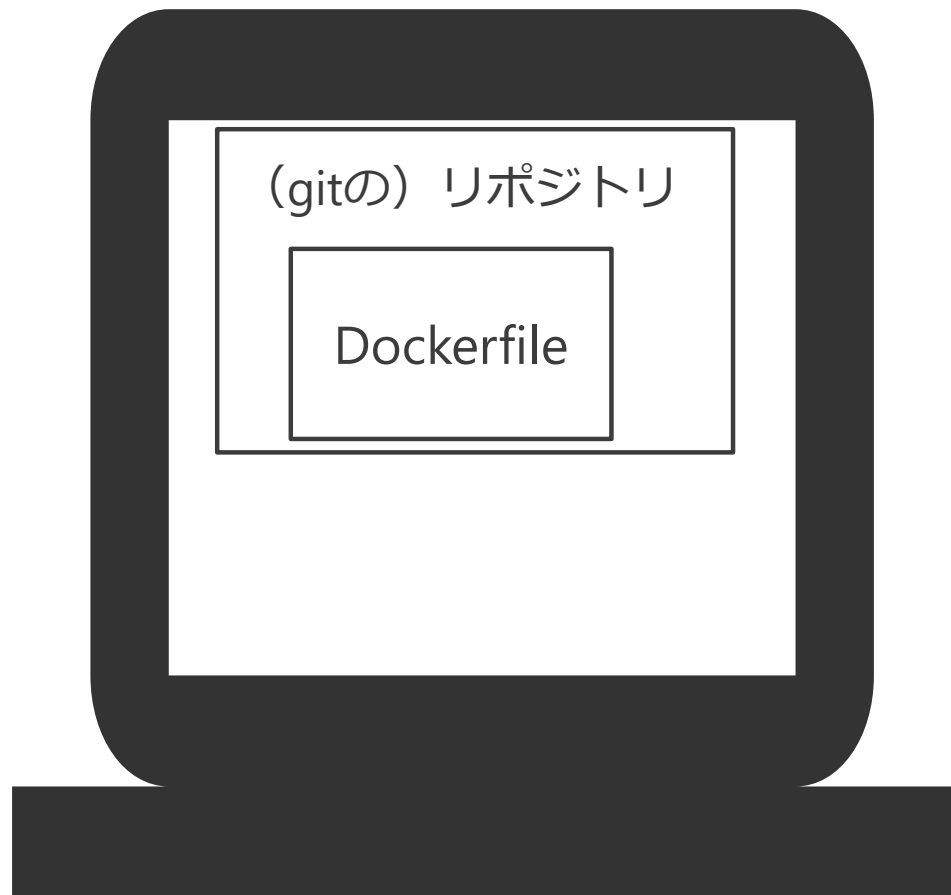
# *dockerhubとgithubの連携*

## ②githubとdockerhubを連携



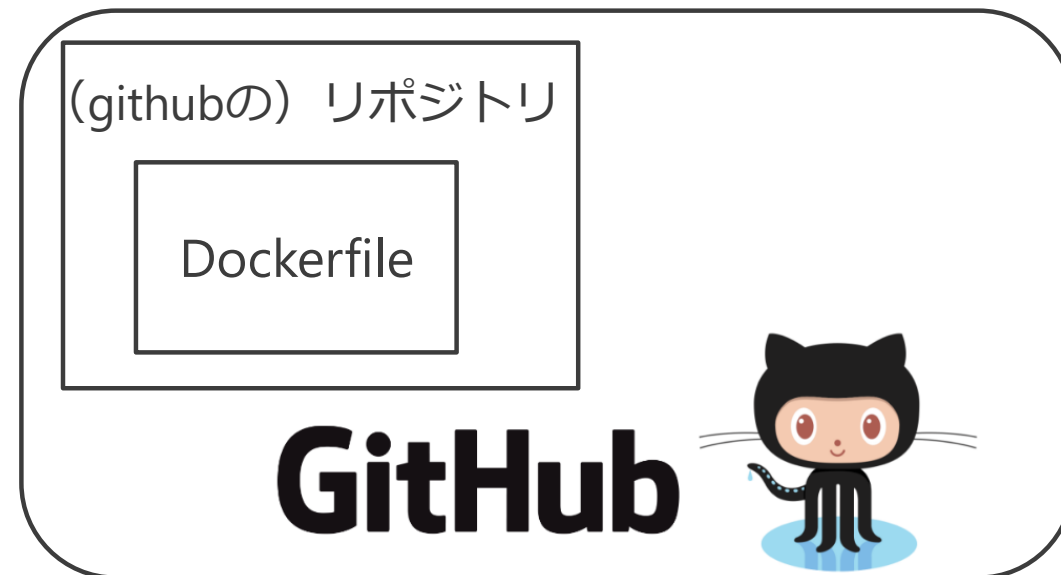
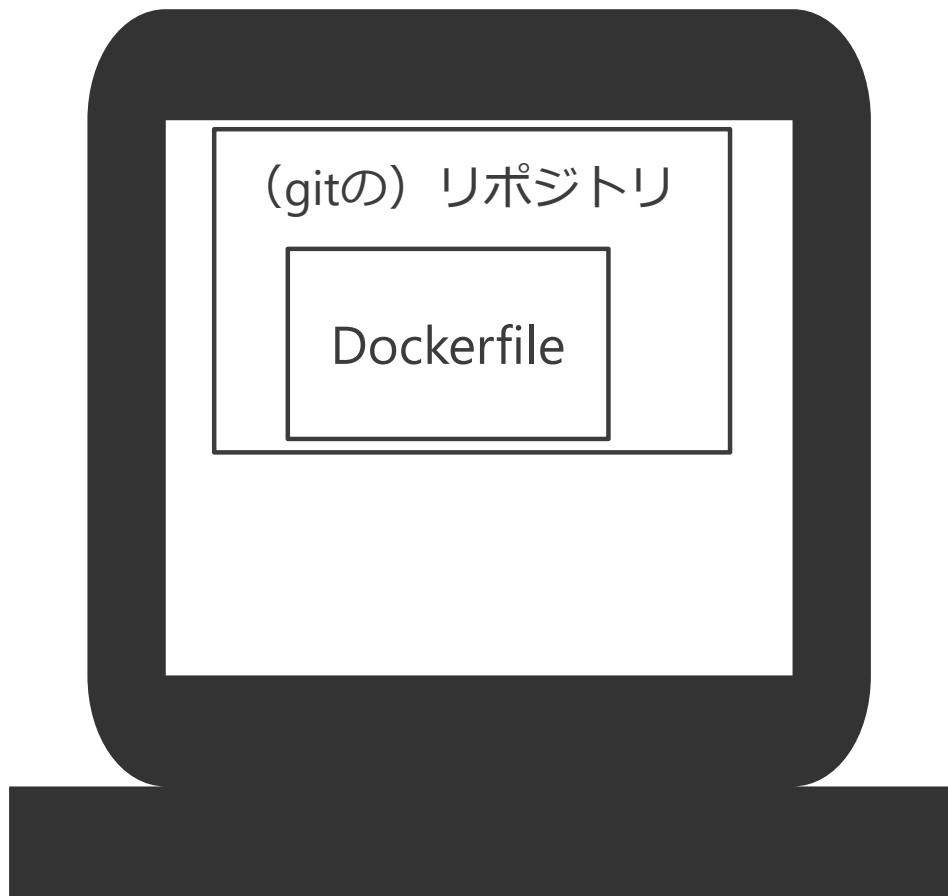
# *dockerhubとgithubの連携*

## ②githubとdockerhubを連携



# *dockerhubとgithubの連携*

## ③docker imageを作成

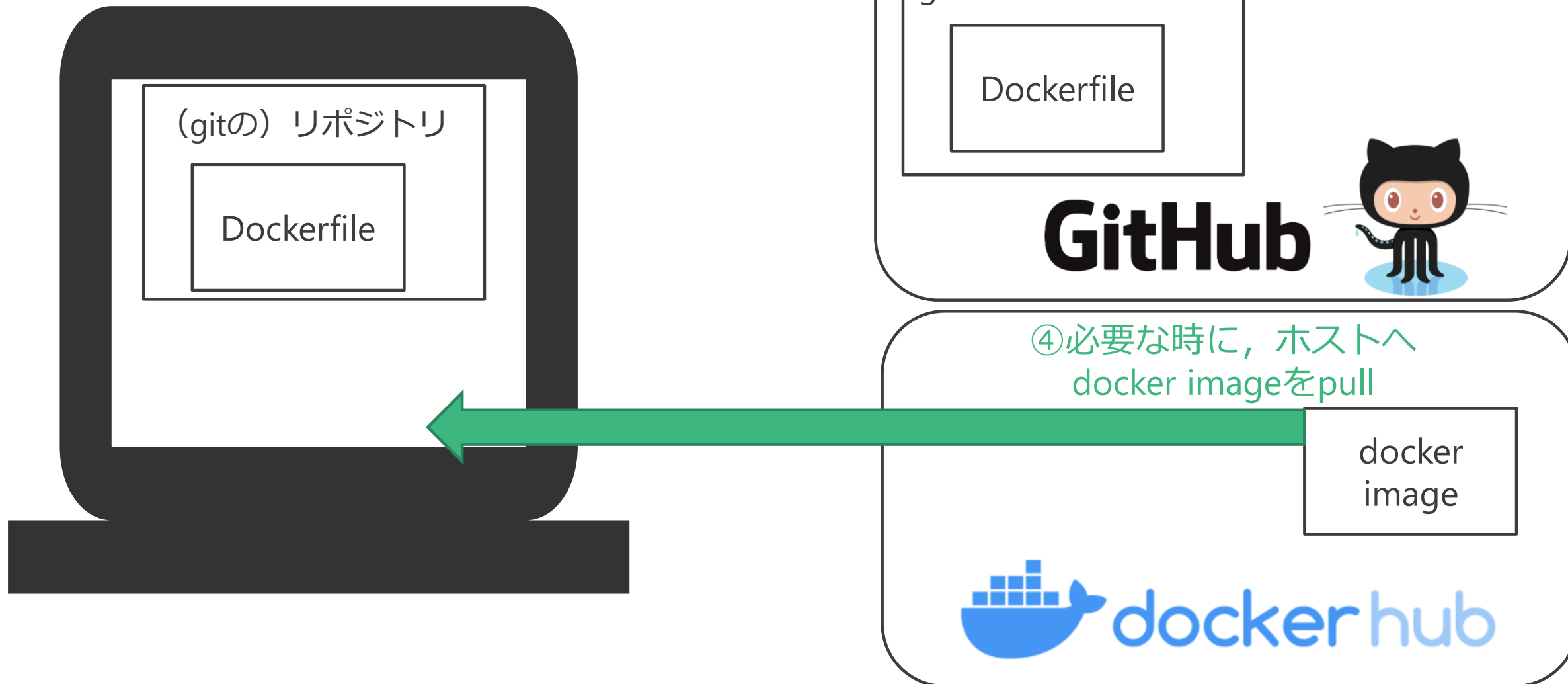


③dockerhubでDockerfileをbuildして  
docker imageを作成（自動build）



# *dockerhub*と*github*の連携

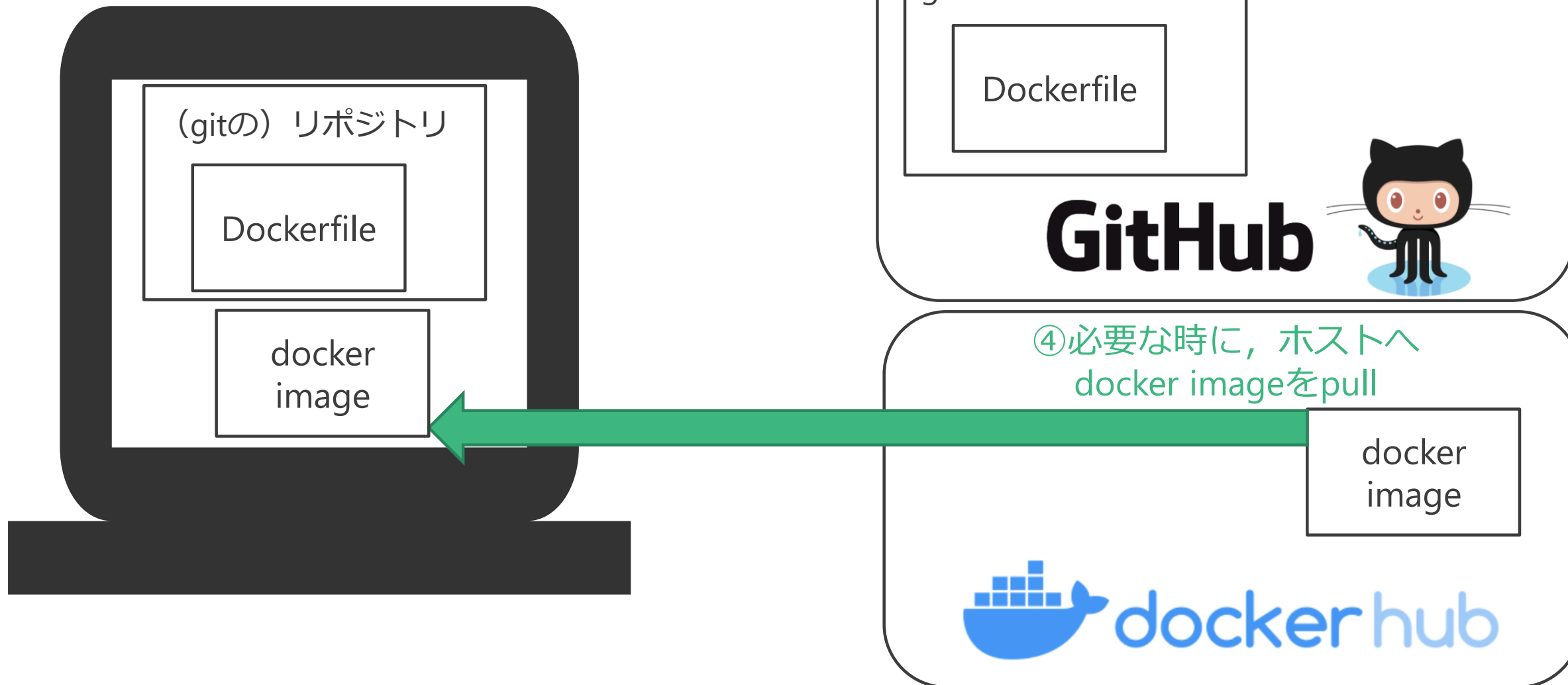
④ (必要な時に) docker imageをpull





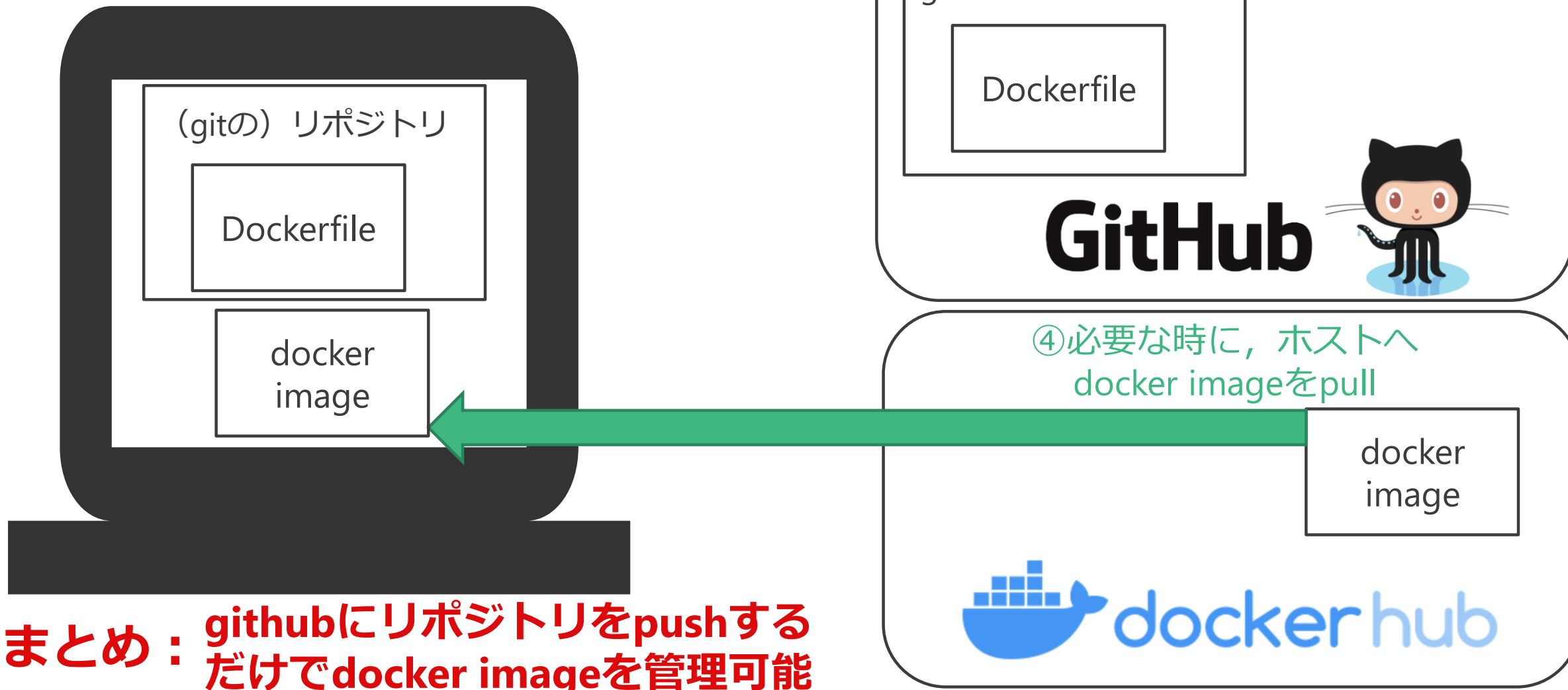
# *dockerhubとgithubの連携*

④ (必要な時に) docker imageをpull

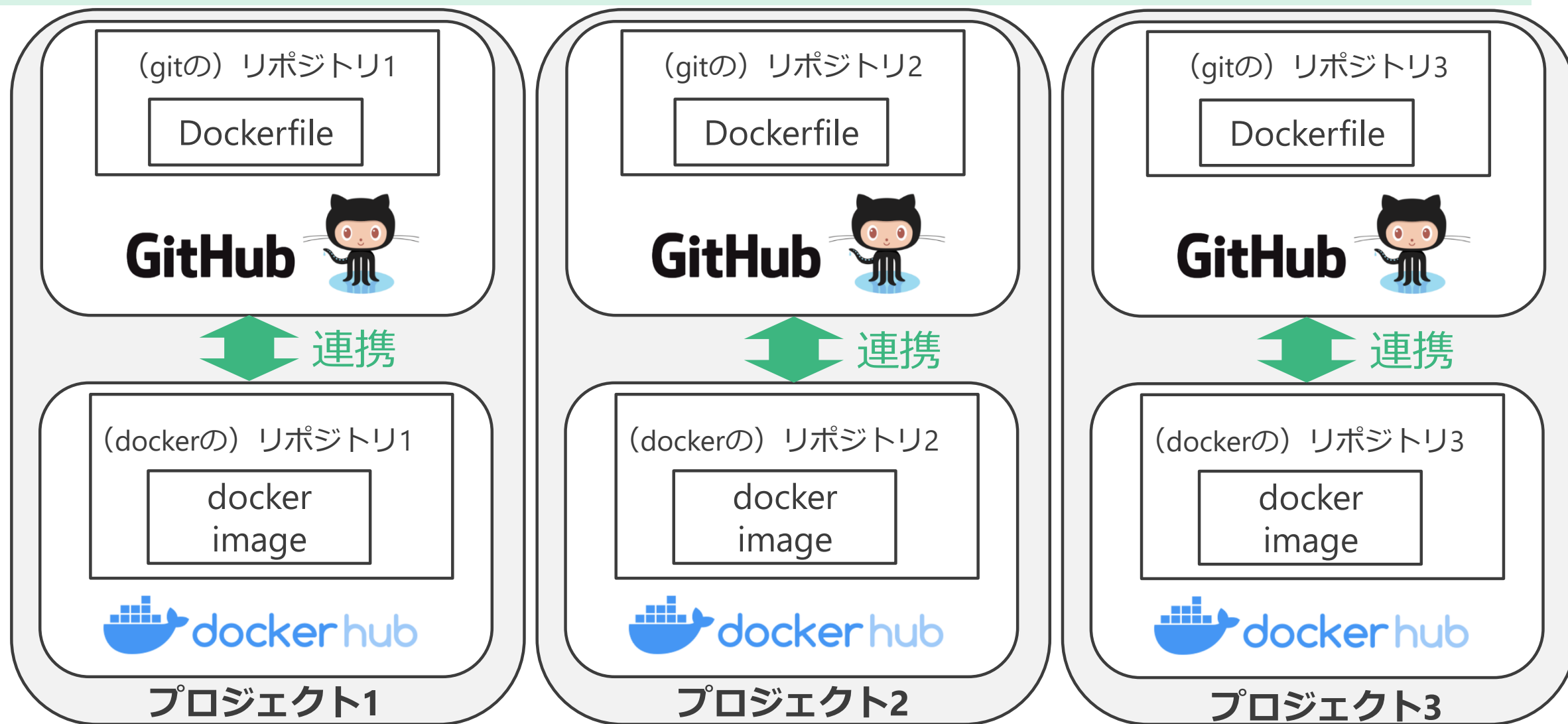


# *dockerhubとgithubの連携*

④ (必要な時に) docker imageをpull



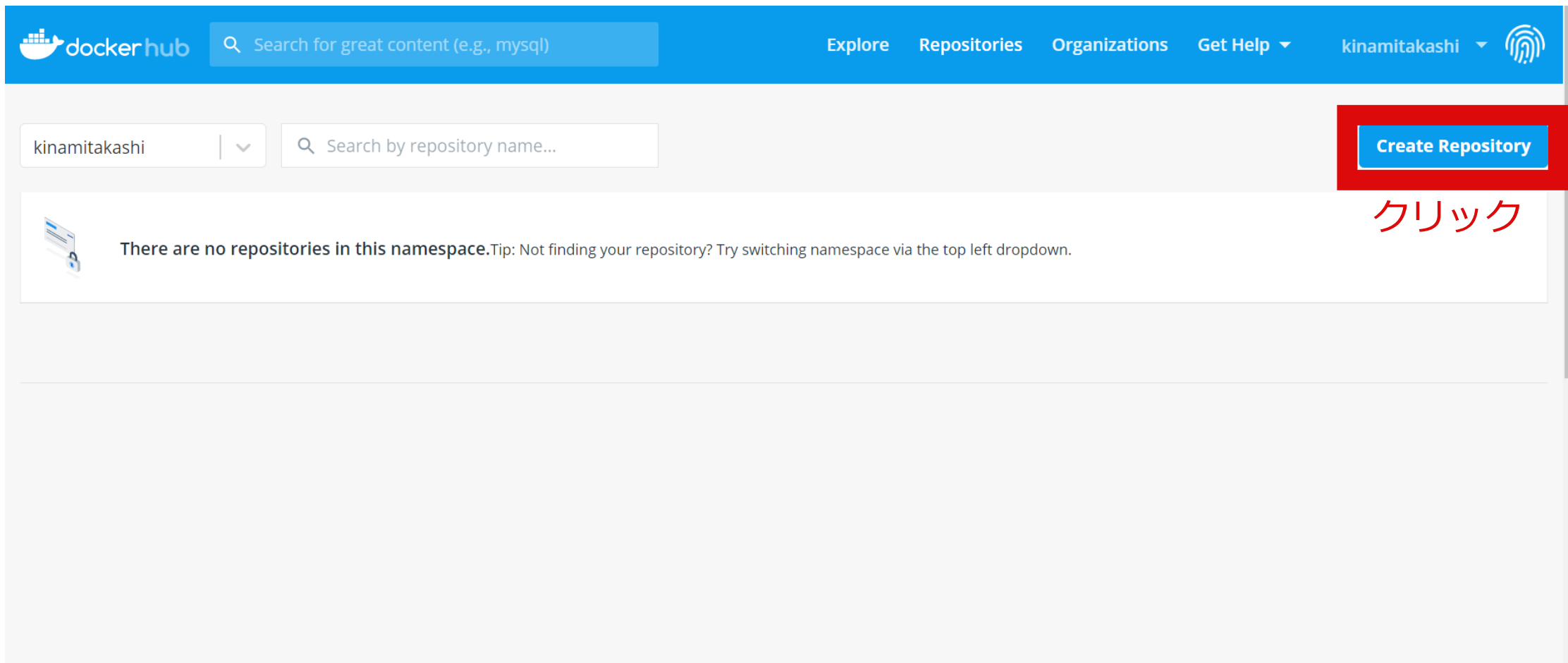
# 各プロジェクトごとで連携



各プロジェクトごとに、github・dockerhubでリポジトリを作成，連携させる

# リポジトリ作成

リポジトリを作成する（既に作成してある場合はそれを用いても良い）



The screenshot shows the Docker Hub website interface. At the top, there is a blue navigation bar with the Docker Hub logo, a search bar, and links for Explore, Repositories, Organizations, Get Help, and a user profile for kinamitakashi. Below the navigation bar, there is a section for the 'kinamitakashi' namespace. It includes a dropdown menu for the namespace and a search bar for repository names. A red box highlights the 'Create Repository' button, with the Japanese text 'クリック' (Click) written below it. The main content area displays a message: 'There are no repositories in this namespace. Tip: Not finding your repository? Try switching namespace via the top left dropdown.'

# リポジトリ作成

リポジトリを作成する（既に作成してある場合はそれを用いても良い）

Repositories > Create

Using 0 of 1 private repositories. [Get more](#)

Create Repository **リポジトリ名**

kinamitakashi **my-first-repo**

Description

Visibility

Using 0 of 1 private repositories. [Get more](#)

☒ **Public** Public repositories appear in Docker Hub search results

☐ **Private** Only you can view private repositories

**リポジトリがpublicかprivateか選択  
(無料プランではprivateリポジトリはひとつのみ)**

Build Settings (optional)

Autobuild triggers a new build with every git push to your source code repository. [Learn More](#)

Connected Disconnected

**クリックすると作成完了**

Cancel **Create** Create & Build

Pro tip

You can push a new image to this repository using the CLI

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

Make sure to change *tagname* with your desired image repository tag.

# githubと連携

## githubとdockerhubを連携する

The screenshot shows the Docker Hub interface for a repository named 'kinamitakashi / my-first-repo'. The 'Builds' tab is selected and highlighted with a red box. Below the tab, the 'Automated Builds' section is visible, with a heading 'ENABLE AUTOBUILD BY CONNECTING TO AN EXTERNAL REPOSITORY SOURCE'. Two buttons are shown: 'Link to GitHub' (labeled 'Connected') and 'Link to Bitbucket' (labeled 'Disconnected'). The 'Link to GitHub' button is highlighted with a red box. Below the screenshot, red text reads: 'Buildsを選択' (Select Builds), 'githubと連携させる' (Connect to GitHub), and '(初回は「Disconnected」なので設定の必要あり)' (Initial setup is required as it is 'Disconnected').

Buildsを選択

githubと連携させる  
(初回は「Disconnected」なので設定の必要あり)

## Dockerfileのbuildの設定

The screenshot shows the Docker Hub interface for configuring builds. The 'Build configurations' section includes settings for source repository, build location, autotest, and repository links. The 'BUILD RULES' section is highlighted with a red box and contains a table of build rules.

**Build configurations**

SOURCE REPOSITORY: kinamitakashi / my-first-repo (Builds | Edit)

NOTE: Changing source repository may affect existing build rules.

BUILD LOCATION: Build on Docker Hub's infrastructure

AUTOTEST: ☒ Off, ☐ Internal Pull Requests, ☐ Internal and External Pull Requests

REPOSITORY LINKS: ☒ Off, ☐ Enable for Base Image

**BUILD RULES** +

The build rules below specify how to build your source into Docker images.

Source Type	Source	Docker Tag	Dockerfile location	Build Context	Autobuild	Build Caching	
Branch	master	latest	Dockerfile	/	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Tag	/^[0-9.]+\$/	{sourcerefs}	Dockerfile	/	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

[View example build rules](#)

**BUILD ENVIRONMENT VARIABLES** +

Buttons: Cancel, Save, Save and Build

# githubと連携

## Dockerfileのbuildの設定

The screenshot shows the Docker Hub interface for configuring builds. The 'Build configurations' section is visible, showing the source repository as 'kinamitakashi' and the build location as 'Build on Docker Hub's infrastructure'. The 'AUTOTEST' option is set to 'off'.

デフォルトではBUILD RULESが一つだけなので、下記のBUILD RULESを追加  
→ Source Type 「Tag」, Source 「/<sup>^</sup>[0-9.]+\$/」, Docker Tag 「{sourceref}」  
(このルールを追加するとdocker imageのバージョン管理が可能になる)

The 'BUILD RULES' section is highlighted with a red box. It shows a table of build rules with columns for Source Type, Source, Docker Tag, Dockerfile location, Build Context, Autobuild, and Build Caching. Two rules are listed: one for 'Branch' (master) and one for 'Tag' (/<sup>^</sup>[0-9.]+\$/).

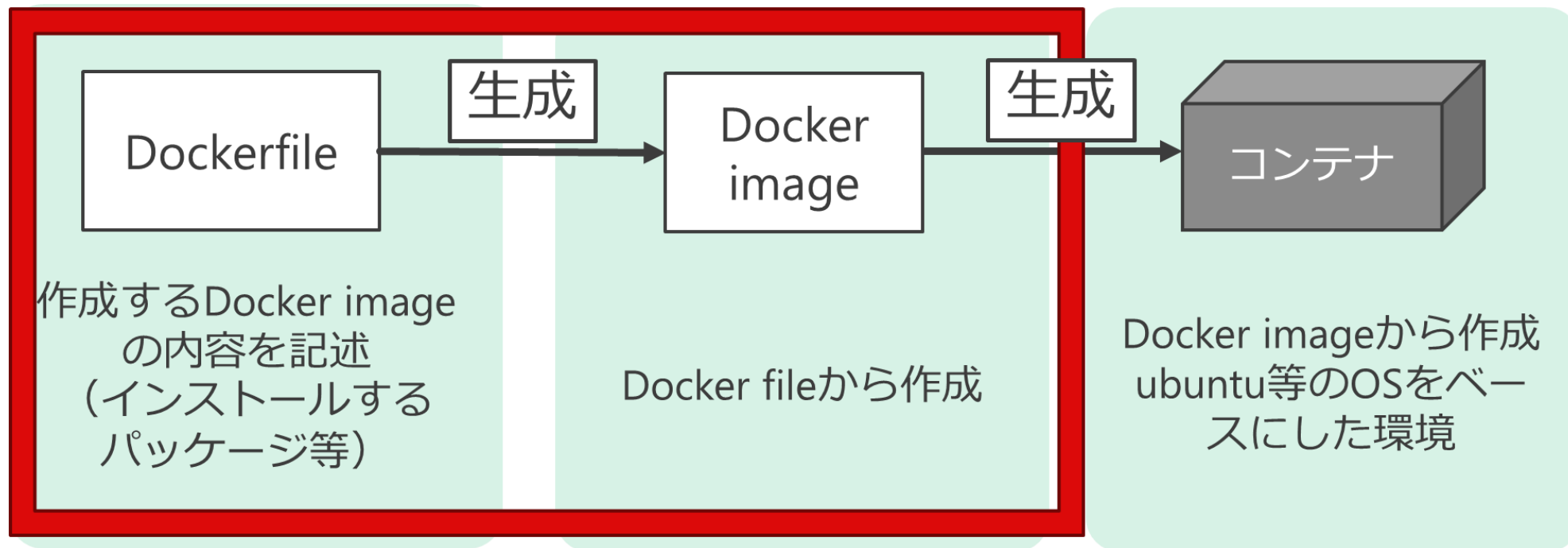
Source Type	Source	Docker Tag	Dockerfile location	Build Context	Autobuild	Build Caching
Branch	master	latest	Dockerfile	/	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Tag	/ <sup>^</sup> [0-9.]+\$/	{sourceref}	Dockerfile	/	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Buttons at the bottom: Cancel, Save, Save and Build.



# Dockerfileの作成とbuild

githubにDockerfileをアップロードする前にホスト側でもbuildできることを確認  
(詳細は「機械学習のためのdocker.pdf」等を確認)



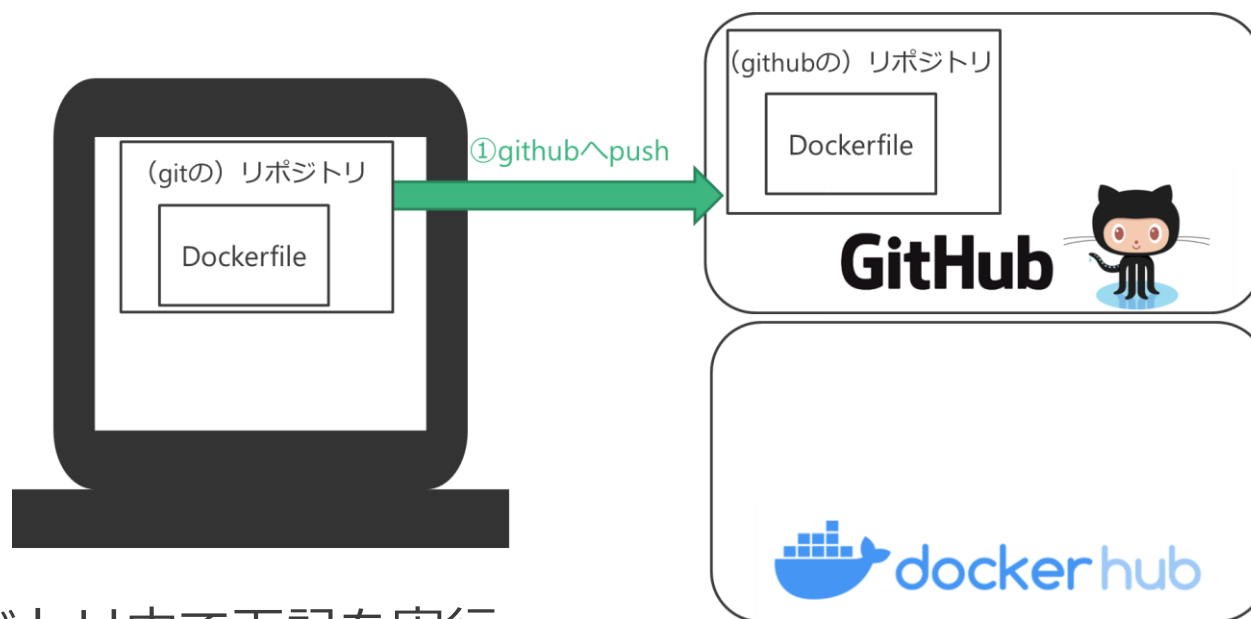
Dockerfileのあるディレクトリ内で下記を実行

```
$ docker build .
```

# githubへpush

githubへリポジトリをpush（Dockerfileをリポジトリの直下に置く）

※ 後述のバージョン管理の関係でDocker関連専用のリポジトリを作成したほうが良いです



Dockerfileがあるリポジトリ内で下記を実行

```
$ git add .
```

```
$ git commit -m "コメント"
```

```
$ git push origin master
```

# dockerhubで自動build

「githubにpushされたリポジトリ内のDockerfile」がdockerhub内で自動build

The screenshot shows the Docker Hub interface for a repository named 'kinamitakashi / my-first-repo'. The 'Builds' tab is selected, showing a 'Build Activity' section with a timeline of the last 2 builds. Below this, the 'Automated Builds' section is visible, showing the 'Latest Build Status' as 'SUCCESS'.

**Build Activity**

Overview of your build activity of the last 2 builds

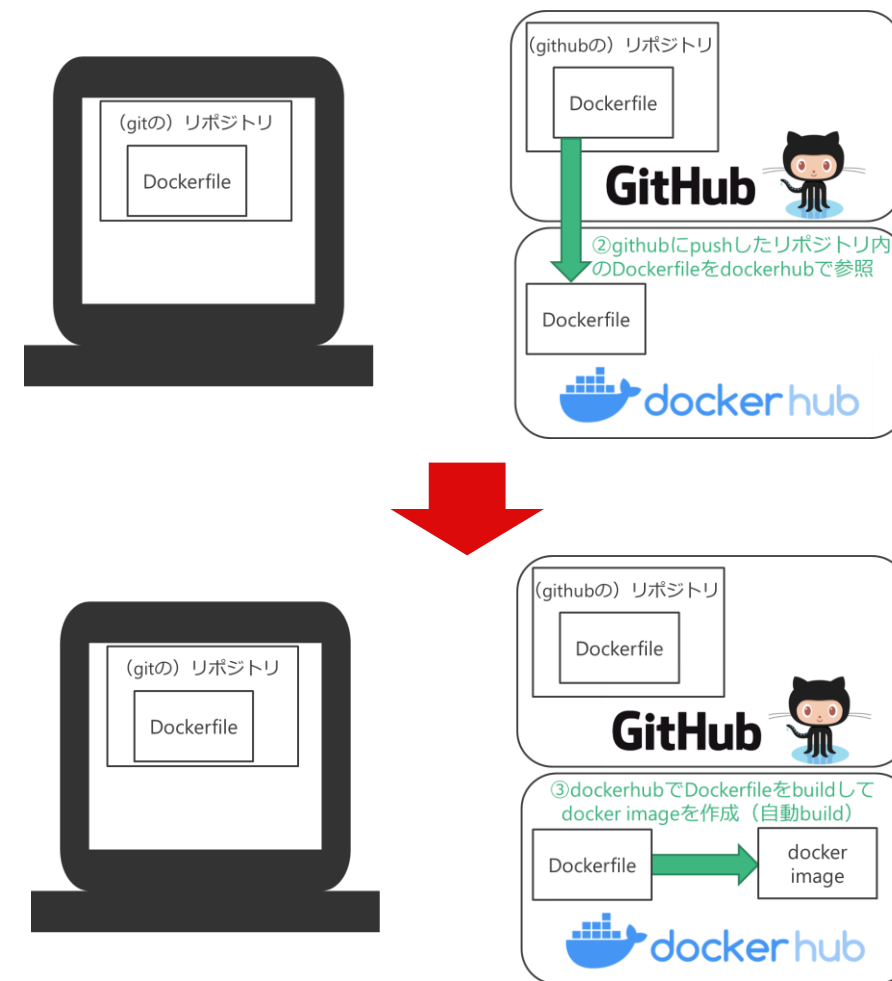
Legend: Queue (blue), Success (green), Failed (red), Canceled (gray)

**Automated Builds**

Autobuild triggers a new build with every git push to your source code repository. [Learn More](#)

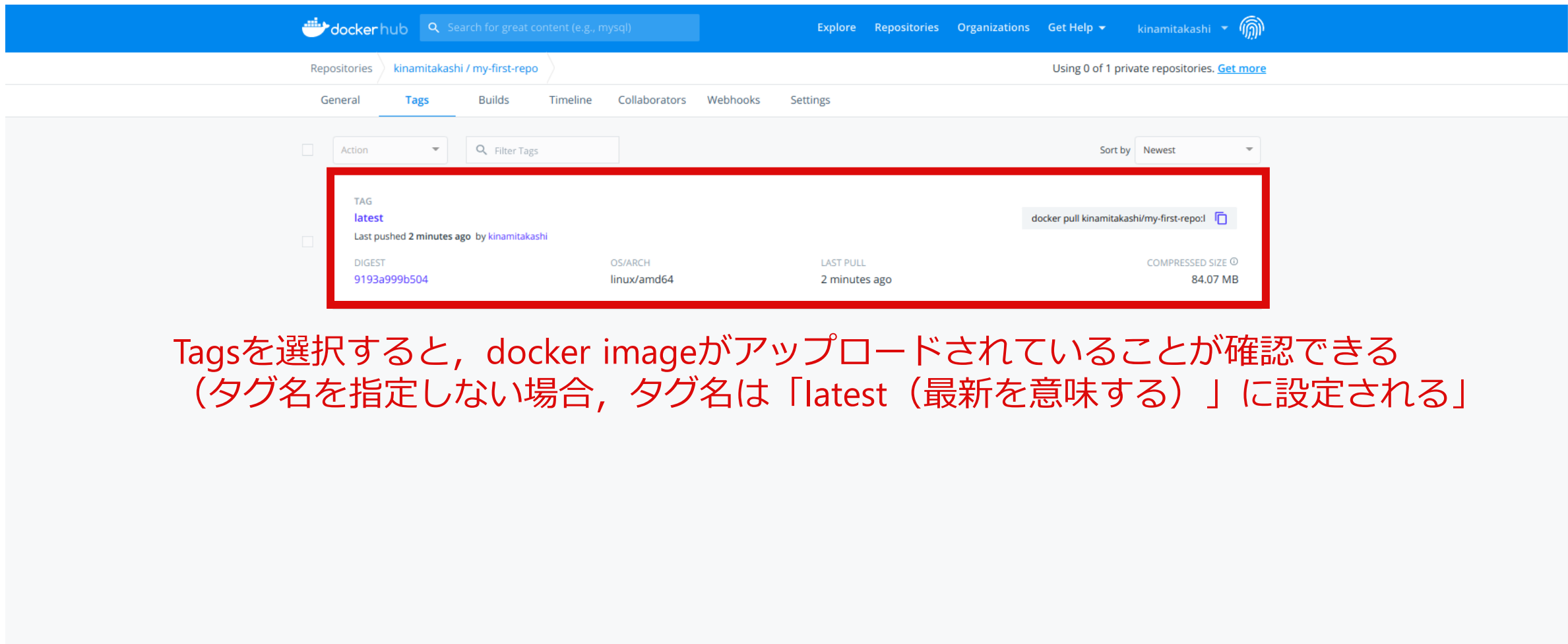
kinami- Latest Build Status が「SUCCESS」になればbuild完了

Docker Tag	Source	Latest Build Status	Autobuild	Build caching
latest	master	SUCCESS	✓	✓
{sourcerefs}	/^[0-9.]+\$/	IN PROGRESS	✓	✓



# *docker image*の確認

buildが完了するとdocker imageがdockerhubにアップロードされるので確認



The screenshot shows the Docker Hub interface for a repository named 'kinamitakashi / my-first-repo'. The 'Tags' tab is selected, displaying a table of image tags. A red box highlights the 'latest' tag row.

TAG	DIGEST	OS/ARCH	LAST PULL	COMPRESSED SIZE
latest	9193a999b504	linux/amd64	2 minutes ago	84.07 MB

Tagsを選択すると、docker imageがアップロードされていることが確認できる  
(タグ名を指定しない場合、タグ名は「latest（最新を意味する）」に設定される）

# *docker image*のバージョン管理

前ページでアップロードされたdocker imageにタグ付けを行うために  
githubのリポジトリにタグを設定して再push

```
$ git tag -a <タグ> -m "コメント"
```

```
$ git push origin <タグ>
```

タグに1.0を設定した場合（タグ名に指定できるのは「1.0」等の数値のみ）

```
$ git tag -a 1.0 -m "version 1.0 released"
```

```
$ git push origin 1.0
```

# *docker image*のバージョン管理

dockerhubを見ると、指定したタグ名でdocker imageがアップロードされていることが確認できる（[P.19](#)と同様に、buildが完了するまで待つ必要あり）

The screenshot shows the Docker Hub interface for the repository `kinamitakashi / my-first-repo`. The `Tags` tab is selected, displaying a list of image tags. Two tags are visible: `1.0` and `latest`. Each tag entry includes a checkbox, the tag name, the time since it was pushed, the user `kinamitakashi`, the digest, the OS/ARCH (`linux/amd64`), the last pull time, and the compressed size (`84.07 MB`). A `docker pull` command is provided for each tag.

Tag	Last pushed	Digest	OS/ARCH	Last pull	Compressed size
<code>1.0</code>	a minute ago by kinamitakashi	<code>4f7fb1c09d6e</code>	<code>linux/amd64</code>	a minute ago	84.07 MB
<code>latest</code>	6 minutes ago by kinamitakashi	<code>d0e04f1eb081</code>	<code>linux/amd64</code>	6 minutes ago	84.07 MB

# *docker image*のバージョン管理

dockerhubを見ると、指定したタグ名でdocker imageがアップロードされていることが確認できる（[P.19](#)と同様に、buildが完了するまで待つ必要あり）

The screenshot shows the Docker Hub interface for the repository `kinamitakashi / my-first-repo`. The `Tags` tab is selected, displaying a list of image tags. A red box highlights the first tag, `1.0`, which was pushed a minute ago. Below it, the `latest` tag is also shown, pushed 6 minutes ago. Both images are for `linux/amd64` and have a compressed size of 84.07 MB. A red text overlay at the top of the tag list reads: 指定したタグ名のdocker imageがアップロード

TAG	OS/ARCH	LAST PULL	COMPRESSED SIZE
<a href="#">1.0</a> Last pushed a minute ago by kinamitakashi	linux/amd64	a minute ago	84.07 MB
<a href="#">latest</a> Last pushed 6 minutes ago by kinamitakashi	linux/amd64	6 minutes ago	84.07 MB

# まとめ

1. Dockerfileを作成
2. ホスト側でDockerfileをbuild（正しくbuildができるか確認）

```
$ docker build .
```

3. 2.でbuildしたDockerfileが入ったリポジトリをgithubへpush（latestを更新）

```
$ git add .
```

```
$ git commit -m “コメント”
```

```
$ git push origin master
```

4. 3.でpushしたリポジトリをタグ名を指定して再push（バージョン管理）

```
$ git tag -a <タグ（タグ名は「1.0」等の数値のみ）> -m “コメント”
```

```
$ git push origin <タグ（上記コマンドで設定したタグ名）>
```



# 参考資料

---

- ・ Docker HubとGitHubを連携してリポジトリからDockerイメージを自動ビルドする方法

(2020/4/30時点)

<https://kamatimaru.hatenablog.com/entry/2020/04/30/184615>

- ・ git でリモートのタグやブランチを削除する方法

<https://qiita.com/usamik26/items/7e53bae128bf130b8a32>