



POLITECHNIKA
LUBELSKA
WYDZIAŁ ELEKTROTECHNIKI
I INFORMATYKI

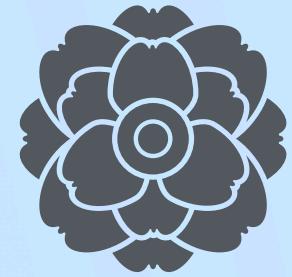
Programowanie Aplikacji w Chmurze Obliczeniowej

Laboratorium nr 10

Konfiguracja i uruchomienie przykładowego łańcucha CI w oparciu o Github oraz usługę Github Actions. Automatyzacja budowania obrazów z wykorzystaniem buldkitd oraz z własnym schematem tagowania

Dr inż. Sławomir Przyłucki
s.przylucki@pollub.pl





Przygotowanie środowiska CI - cz. I

1 Przed rozpoczęciem działań opisanych w tej instrukcji należy zagwarantować, że:

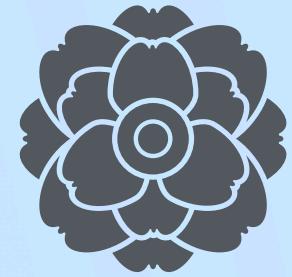
- w wykorzystywanym systemie jest zainstalowane (i skonfigurowane) narzędzie git,
- w wykorzystywanym systemie jest zainstalowane (i skonfigurowane) narzędzie gh
- jesteśmy zalogowani jest w systemie GitHub z użyciem dedykowanego tokenu PAT

```
mac ~ /examples/L10
> gh auth status
github.com
✓ Logged in to github.com account SenatorP51 (keyring)
- Active account: true
- Git operations protocol: ssh
- Token: ghp_*****
- Token scopes: 'admin:gpg_key', 'admin:org', 'admin:public_key', 'admin:ssh_signing_key', 'codespace', 'delete:packages',
'delete_repo', 'project', 'repo', 'user', 'workflow', 'write:packages'
```

2 Do wybranego podkatalogu w katalogu domowym należy rozpakować pliki źródłowe dostępne na moodle, w katalogi do laboratorium 10. Plik: **lab10_examples.zip**



```
mac ~ /examples/L10
> ll -a
drwx-----@ - slawek staff 22 mar 2024 .github
.rwx-----@ 10 slawek staff 11 wrz 2024 .gitignore
.rwx-----@ 263 slawek staff 11 wrz 2024 Dockerfile
.rwx-----@ 11k slawek staff 11 wrz 2024 LICENSE
.rwx-----@ 286 slawek staff 11 wrz 2024 README.md
drwx-----@ - slawek staff 22 mar 2024 src
```



Przygotowanie środowiska CI - cz. II

3

Należy zainicjować repo git

```
❯ git init -b main
Zainicjowano puste repozytorium Gita w /Users/slawek/examples/L10/.git/
❯
```

4

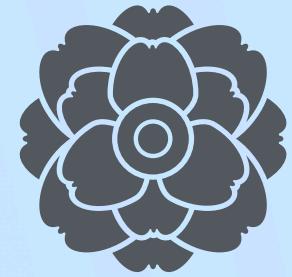
Na Github należy utworzyć repozytorium na podstawie repo z punktu 3 (klonowanie)

```
❯ gh repo create
? What would you like to do? Push an existing local repository to GitHub
? Path to local repository .
? Repository name lab10
? Description Repozytorium dedykowane przykładowi z lab 10
? Visibility Public
✓ Created repository SenatorP51/lab10 on GitHub
https://github.com/SenatorP51/lab10
? Add a remote? Yes
? What should the new remote be called? origin
✓ Added remote git@github.com:SenatorP51/lab10.git
```

5

Sprawdzenie utworzenia repo na Github

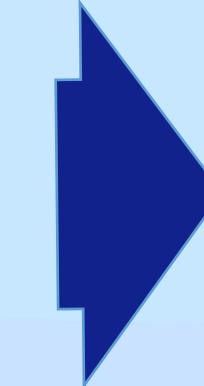
```
❯ gh repo list | grep lab10
SenatorP51/lab10          Repozytorium dedykowane przykładowi z lab 10    public
```



Przygotowanie środowiska CI - cz. III

6

Przygotowanie do pierwszego commit-a:



```
~/examples/L10 🌐 main [!]  
❯ git add .  
~/examples/L10 🌐 main [++(6)]  
❯ git status  
Na gałęzi main
```

Jeszcze nie ma zapisów

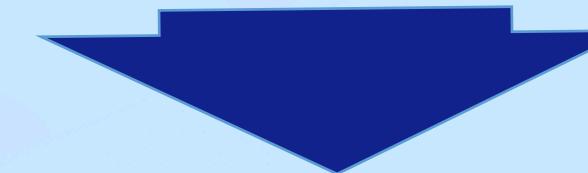
Zmiany do złożenia:

(użyj „git rm --cached <plik>...”, aby wycofać)

nowy plik:	.github/workflows/gha_example.yml
nowy plik:	.gitignore
nowy plik:	Dockerfile
nowy plik:	LICENSE
nowy plik:	README.md
nowy plik:	src/index.html

7

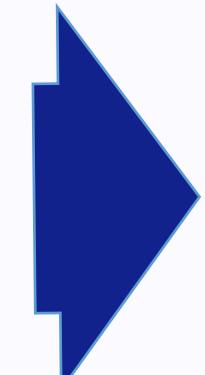
Wykonanie pierwszego commit-a:



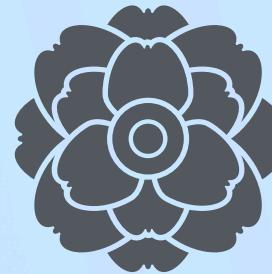
```
~/examples/L10 🌐 main [++(6)]  
❯ git commit -m "Inicjalizacja repo lab10"  
[main (zapis-korzeń) e6dd510] Inicjalizacja repo lab10  
6 files changed, 295 insertions(+)  
create mode 100755 .github/workflows/gha_example.yml  
create mode 100755 .gitignore  
create mode 100755 Dockerfile  
create mode 100755 LICENSE  
create mode 100755 README.md  
create mode 100755 src/index.html
```

8

Synchronizacja z repo na Github:



```
~/examples/L10 🌐 main  
❯ git push -u origin main  
Wymienianie obiektów: 11, gotowe.  
Zliczanie obiektów: 100% (11/11), gotowe.  
Kompresja delt z użyciem do 10 wątków  
Kompresowanie obiektów: 100% (7/7), gotowe.  
Zapisywanie obiektów: 100% (11/11), 5.53 KiB | 2.77 MiB/s, gotowe.  
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
To github.com:SenatorP51/lab10.git  
 * [new branch]      main -> main  
branch 'main' set up to track 'origin/main'.
```



PAwChO – Laboratorium 10

Przygotowanie środowiska CI – cz. IV

9

Należy sprawdzić czy na Github zostało poprawnie utworzone repozytorium *lab 9*

10

W ramach udostępnionych źródeł dostępny jest przykładowy łańcuch CI dla usługi GitHub Actions

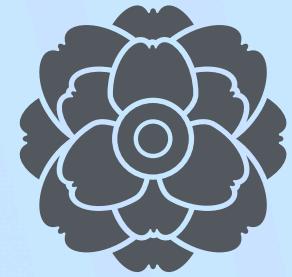


Przydatne linki do analizy i samodzielnego tworzenia łańcuchów CI/CD w oparciu o Github Actions

<https://docs.github.com/en/actions/quickstart>

<https://github.com/actions/starter-workflows>

```
EKSPLORATOR ...  
└ L10  
   └ .github  
      └ workflows  
         └ gha_example.yml 5  
            > src  
            ⚡ .gitignore  
            📜 Dockerfile  
            📄 LICENSE  
            ⓘ README.md  
.  
gha_example.yml 5 x  
.github > workflows > gha_example.yml  
1 name: GHACTION example  
2  
3 on:  
4   workflow_dispatch:  
5     push:  
6       tags:  
7         - 'v*'  
8  
9 jobs:  
10  ci_step:  
11    name: Build, tag and push Docker image to DockerHub  
12    runs-on: ubuntu-latest  
13  
14 steps:  
15  -  
16    name: Check out the source_repo  
17    uses: actions/checkout@v4  
18  
19  
20  -  
21    name: Docker metadata definitions  
22    id: meta  
23    uses: docker/metadata-action@v5  
24    with:  
25      images: ${{ vars.DOCKERHUB_USERNAME }}/example  
26      flavor: latest=false  
27      tags: |  
28        type=sha,priority=100,prefix=sha-,format=short  
        type=semver,priority=200,pattern={{version}}  
29  
30  
31  -  
32    name: QEMU set-up  
33    uses: docker/setup-qemu-action@v3  
34  
35  
36  -  
37    name: Buildx set-up  
38    uses: docker/setup-buildx-action@v3  
39  
40  
41  -  
42    name: Login to DockerHub  
43    uses: docker/login-action@v3  
44    with:  
45      username: ${{ vars.DOCKERHUB_USERNAME }}  
46      password: ${{ secrets.DOCKERHUB_TOKEN }}  
47  
48  
49  -  
50    name: Build and push Docker image  
51    uses: docker/build-push-action@v5  
52    with:  
53      context: .  
54      file: ./Dockerfile  
55      platforms: linux/amd64,linux/arm64  
56      push: true  
57      cache-from: |  
        type=registry,ref=${{ vars.DOCKERHUB_USERNAME }}/example:cache  
      cache-to: |  
        type=registry,ref=${{ vars.DOCKERHUB_USERNAME }}/example:cache  
      tags: ${{ steps.meta.outputs.tags }}
```



PAwChO – Laboratorium 10

Łańcuch CI - Checkout

Ten moduł **Action** sprawdza dane repozytorium zdefiniowane przez zmienną `$GITHUB_WORKSPACE`, aby potwierdzić że uruchamiany łańcuch może uzyskać do niego dostęp na zasadach określonych w konfiguracji modułu.

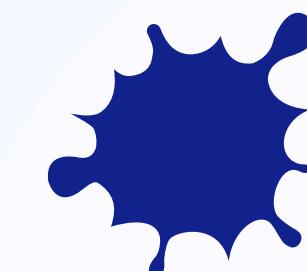
Scenarios

- [Fetch only the root files](#)
- [Fetch only the root files and .github and src folder](#)
- [Fetch only a single file](#)
- [Fetch all history for all tags and branches](#)
- [Checkout a different branch](#)
- [Checkout HEAD[^]](#)
- [Checkout multiple repos \(side by side\)](#)
- [Checkout multiple repos \(nested\)](#)
- [Checkout multiple repos \(private\)](#)
- [Checkout pull request HEAD commit instead of merge commit](#)
- [Checkout pull request on closed event](#)
- [Push a commit using the built-in token](#)

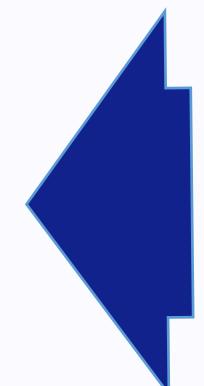
Checkout V4

-
name: Check out the source_repo
uses: [actions/checkout@v4](#)

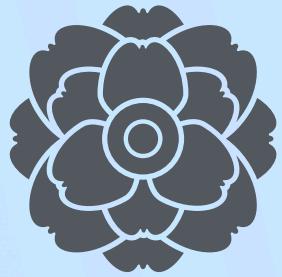
<https://github.com/marketplace/actions/checkout>



W konfiguracji można określić jak zamierza się korzystać z repo - tzw. **Dfefinicja scenariuszy**



<https://github.com/marketplace/actions/checkout#scenarios>



PAwChO – Laboratorium 10

Łańcuch CI - Logowanie - cz. I

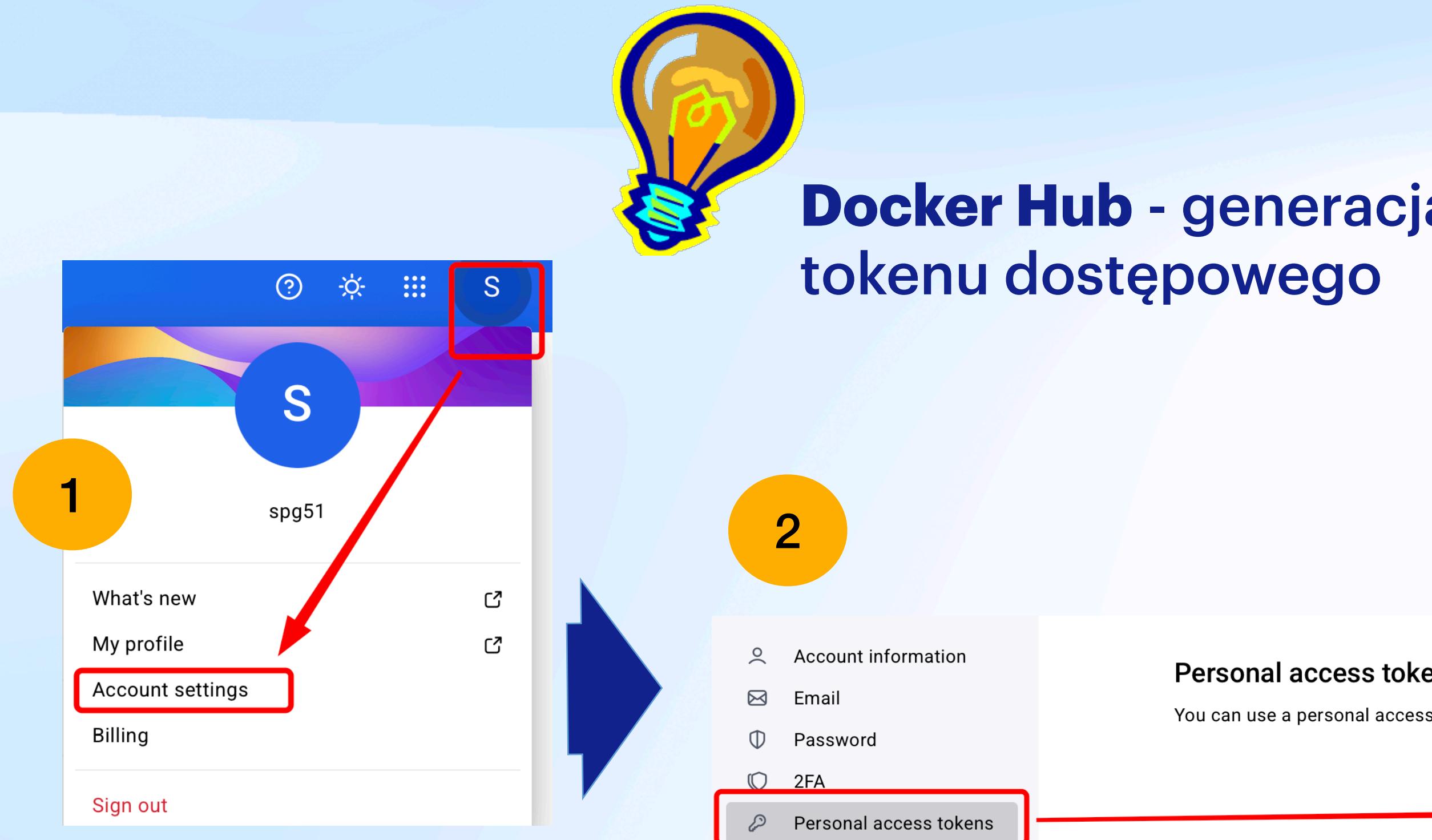
GitHub Action
Docker Login
v3.1.0 Latest version

release v3.1.0 marketplace docker-login ci passing test passing coverage 72%

<https://github.com/marketplace/actions/docker-login>

```
- name: Login to DockerHub
  uses: docker/login-action@v3
  with:
    username: ${{ vars.DOCKERHUB_USERNAME }}
    password: ${{ secrets.DOCKERHUB_TOKEN }}
```

Moduł Action docker/login-action pozwala na zalogowanie się w wielu repozytoriach obrazów.
PROSĘ KONIECZNIE zapoznać się z dokumentacją



Create access token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. [Learn more ↗](#)

Access token description 3

Expiration date

Optional

Access permissions

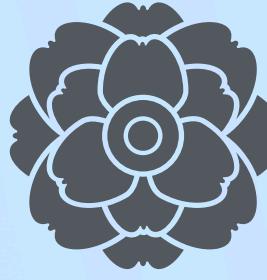
Read, Write, Delete tokens allow you to manage your repositories.

Cancel Generate

Personal access tokens

You can use a personal access token instead of a password for Docker CLI authentication. Create multiple tokens, control their scope, and delete tokens at any time. [Learn more ↗](#)

Generate new token



PAwChO – Laboratorium 10

Łańcuch CI - Logowanie - cz. II



4

Copy access token

Use this token as a password when you sign in from the Docker CLI client. [Learn more ↗](#)

Make sure you copy your personal access token now. Your personal access token is only displayed once. It isn't stored and can't be retrieved later.

Access token description

GA_ACCESS_TOKEN_L10

Expires on

Never

Access permissions

Read, Write, Delete

To use the access token from your Docker CLI client:

1. Run

```
$ docker login -u spg51
```

Copy

2. At the password prompt, enter the personal access token.

```
dckr_pat_███████████
```

Copy

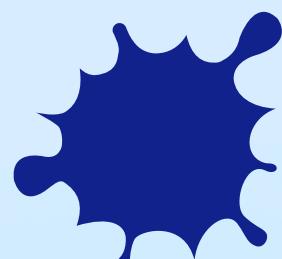
[Back to access tokens](#)

**UWAGA: !!!!!!!! Podobnie jak z PAT na GitHub
należy zapisać wygenerowany token
w bezpiecznym miejscu (np. plik w katalogu
~/.ssh) oraz zapewnić, że nie pozostanie on
na komputerze uczelnianym po zakończeniu
zajęć !!!!!!**

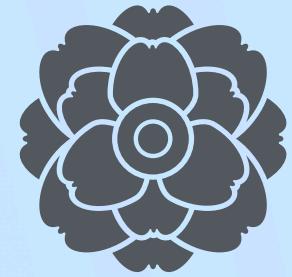
5

Należy w utworzonym repo na Github zdefiniować odpowiednio: zmieną oraz sekret

DOCKERHUB_USERNAME
DOCKERHUB_TOKEN



Zasięg wygenerowanego tokena dostępowego odnosi się do wszystkich repozytoriów (całego konta na DockerHub)



PAwChO – Laboratorium 10

Łańcuch CI - Logowanie - cz. III

Działania na utworzonym repo *lab10*

Należy wkleić utworzony na DockerHub token dostępowy PAT

1A

```
~/examples/L10 🐧 main [✓]
> gh secret set DOCKERHUB_TOKEN
? Paste your secret: ****
✓ Set Actions secret DOCKERHUB_TOKEN for SenatorP51/lab10
```

1B Weryfikacja utworzenia secret-u w interfejsie web Github

The screenshot shows the 'Repository secrets' section of a GitHub repository. On the left, there's a sidebar with links: Security, Code Security, Deploy keys, Secrets and variables (which is highlighted with a green border), Actions (also highlighted with a green border), Codespaces, and Dependabot. The main area shows a table with one row: Name: DOCKERHUB_TOKEN, Last updated: 2 minutes ago, with edit and delete icons. A green arrow points from the 'Actions' link in the sidebar to the 'Actions' button in the top right of the table.

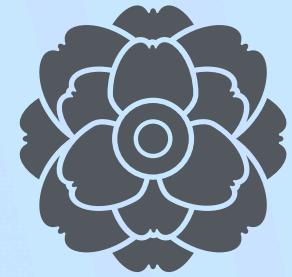
Należy podać własną nazwę użytkownika na Dockerhub

2A

```
~/examples/L10 🐧 main [✓]
> gh variable set DOCKERHUB_USERNAME
? Paste your variable spg51
✓ Created variable DOCKERHUB_USERNAME for SenatorP51/lab10
```

2B Weryfikacja utworzenia zmiennej w interfejsie web Github

The screenshot shows the 'Repository variables' section of a GitHub repository. The sidebar is identical to the previous screenshot. The main area shows a table with one row: Name: DOCKERHUB_USERNAME, Value: spg51, Last updated: 2 minutes ago, with edit and delete icons. A green arrow points from the 'Actions' link in the sidebar to the 'Actions' button in the top right of the table.



PAwChO – Laboratorium 10

Łańcuch CI - Deklarowanie środowiska budowanie obrazów

Marketplace / Actions / Docker Setup Buildx

 GitHub Action
Docker Setup Buildx
v3.0.0 [Latest version](#)

release v3.0.0 | marketplace docker-setup-buildx | ci passing | test passing | coverage 40%

About
GitHub Action to set up Docker [Buildx](#).

Marketplace / Actions / Docker Setup QEMU

 GitHub Action
Docker Setup QEMU
v3.0.0 [Latest version](#)

release v3.0.0 | marketplace docker-setup-qemu | ci passing | test passing | coverage 100%

About
GitHub Action to install [QEMU](#) static binaries.

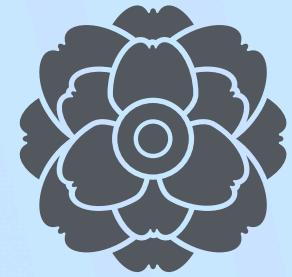
<https://github.com/marketplace/actions/docker-setup-buildx>

<https://github.com/marketplace/actions/docker-setup-qemu>

Powyższe Actions są zwykle stosowane łącznie. Pozwalają na danym hoście runner-a zainstalować wszystkie niezbędne komponenty do budowania obrazu z wykorzystaniem silnika Buildkit.



Bez instalacji tych składników nie możliwe jest budowanie obrazów na wiele architektur sprzętowych ani wykorzystanie rozproszonego cache-a.



PAwChO – Laboratorium 10

Łańcuch CI - Tagowanie obrazów - cz. I

Marketplace / Actions / Docker Metadata action

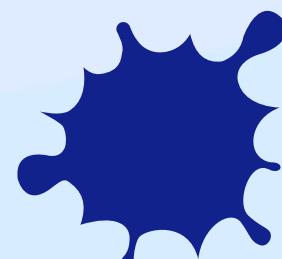


release v5.7.0 | marketplace docker-metadata-action | ci passing | test passing | coverage 93%

About

GitHub Action to extract metadata from Git reference and GitHub events. This action is particularly useful if used with [Docker Build Push](#) action to tag and label Docker images.

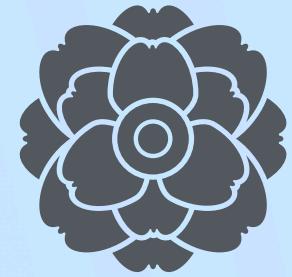
<https://github.com/marketplace/actions/docker-metadata-action>



Id: <nazwa> nie jest bezpośrednio związana z konfiguracją tego modułu Actions. Jest to metoda przypisywania nazwy do danego kroku (danego step-u) aby można było w prosty sposób odwoływać się do niego w innych miejscach łańcucha CI

```
- name: Docker metadata definitions
  id: meta
  uses: docker/metadata-action@v5
  with:
    images: ${{ vars.DOCKERHUB_USERNAME }}/example2
    flavor: latest=false
    tags:
      - type=sha,priority=100,prefix=sha-,format=short
      - type=semver,priority=200,pattern={{version}}
```

Deklaracja nazwy obrazu (bez tag-u): docker.io/<dockerhub_user>/<user_repo_name>



Łańcuch CI - Tagowanie obrazów - cz. II

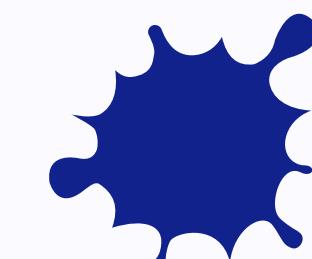
```
-  
  name: Docker metadata definitions  
  id: meta  
  uses: docker/metadata-action@v5  
  with:  
    images: ${{ vars.DOCKERHUB_USERNAME }}/example2  
    flavor: latest=false  
    tags: |  
      type=sha,priority=100,prefix=sha-,format=short  
      type=semver,priority=200,pattern={{version}}
```

Domyślnie, obraz jest tag-owany zgodnie z przyjętym schematem + generowany jest tag `latest`. Wartość `latest=false` wyłącza generowanie dodatkowego tag-u `latest`

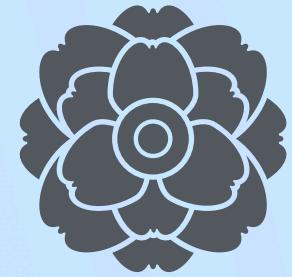
Metadata Action pozwala konfigurować tagowanie obrazów według trzech schematów:

- [Usage](#)
 - [Basic](#)
 - [Semver](#)
 - [Bake definition](#)

Event	Ref	Docker Tags
pull_request	refs/pull/2/merge	pr-2
push	refs/heads/master	master
push	refs/heads/releases/v1	releases-v1
push tag	refs/tags/v1.2.3	1.2.3, 1.2, latest
push tag	refs/tags/v2.0.8-beta.67	2.0.8-beta.67



Każdy schemat pozwala na wygenerowanie określonego tag-u w zależności od użytego trigger-a (od rodzaju event-u)



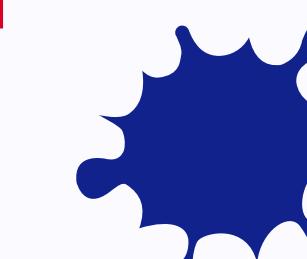
Łańcuch CI - Tagowanie obrazów - cz. III

```
on:  
  workflow_dispatch:  
  push:  
    tags:  
      - 'v*'
```

Zdeklarowane
zdarzenia (event-y)

Schemat sha (hash commit-a) zareaguje tak na event „dispatch” jak i „push”. W przypadku event-u „push” wyższy priorytet ma semver i on będzie tag-ował obraz.

Schemat semver „nie reaguje” na event dispatch, jedynie zareaguje na push tag-a

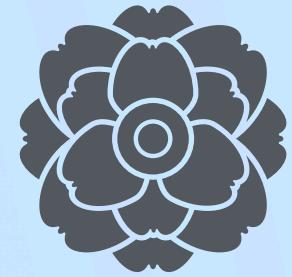


```
-  
  name: Docker metadata definitions  
  id: meta  
  uses: docker/metadata-action@v5  
  with:  
    images: ${{ vars.DOCKERHUB_USERNAME }}/example2  
    flavor: latest=false  
    tags: |  
      type=sha,priority=100,prefix=sha-,format=short  
      type=semver,priority=200,pattern={{version}}
```

Tagowanie zależy od rodzaju eventu a dodatkowo można poszczególnym schematom tag-owania przypisać priorytety. Im wyższy priorytet tym „ważniejszy” schemat.

Domyślna konfiguracja posiada przypisane wartości priorytetów, które można mienić jak wyżej.

<https://github.com/marketplace/actions/docker-metadata-action#priority-attribute>



PAwChO – Laboratorium 10

Łańcuch CI - Budowanie obrazów - cz. I

Marketplace / Actions / Build and push Docker images

Build and push Docker images Actions

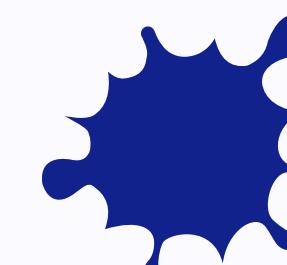
release v6.15.0 marketplace build-and-push-docker-images ci passing test passing coverage 0%

About

GitHub Action to build and push Docker images with [Buildx](#) with full support of the features provided by [Moby BuildKit](#) builder toolkit. This includes multi-platform build, secrets, remote cache, etc. and different builder deployment/namespacing options.

<https://github.com/marketplace/actions/build-and-push-docker-images>

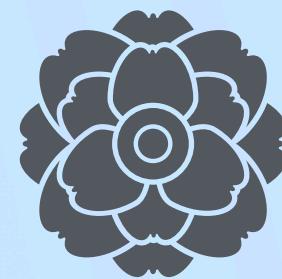
Definicja docelowych architektur sprzętowych (analogicznie jak przy poleceniu docker buildx ...)



```
name: Build and push Docker image
uses: docker/build-push-action@v3
with:
  context: .
  file: ./Dockerfile
  platforms: linux/amd64,linux/arm64
  push: true
  cache-from: |
    type=registry,ref=${{ vars.DOCKERHUB_USERNAME }}/example2:cache
  cache-to: |
    type=registry,ref=${{ vars.DOCKERHUB_USERNAME }}/example2:cache
  tags: ${{ steps.meta.outputs.tags }}
```

Definicja parametrów budowania obrazu
(analogicznie jak przy poleceniu docker build ...)

W przypadku domyślnej nazwy pliku Dockerfile - wartość file: można pominąć



Łańcuch CI - Budowanie obrazów - cz. II

```
name: Build and push Docker image
uses: docker/build-push-action@v3
with:
  context: .
  file: ./Dockerfile
  platforms: linux/amd64,linux/arm64
  push: true
  cache-from: |
    type=registry,ref=${{ vars.DOCKERHUB_USERNAME }}/example2:cache
  cache-to: |
    type=registry,ref=${{ vars.DOCKERHUB_USERNAME }}/example2:cache
  tags: ${{ steps.meta.outputs.tags }}
```

Przesyłanie i pobieranie danych cache, które zgodnie z informacjami z poprzednich wykładów, pozwalają na przyśpieszenie operacji budowania kolejnych wersji obrazu danej aplikacji.



Proszę KONIECZNIE zapoznać się z przykładami z dokumentacji, szczególnie z typami cache oraz sposobami korzystania z każdego z nich.

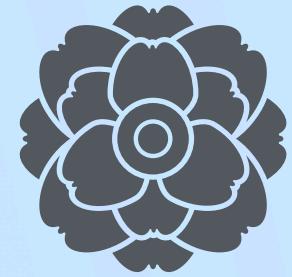
Pobranie z kroku „meta” wygenerowanych tag-ów i przypisanie ich do budowanego obrazu

<https://docs.docker.com/build/ci/github-actions/cache/>



Ponownie POLECAM bardzo przydatną dokumentację i przykłady wykorzystania GitHub Actions w połączeniu ze środowiskiem Docker

<https://docs.docker.com/build/ci/github-actions/>



PAwChO – Laboratorium 10

Github Actions + gh CLI - cz. I

```
~/Labs/lab9 🐫 main [✓]
> gh help actions
Welcome to GitHub Actions on the command line.

GitHub CLI integrates with GitHub Actions to help you manage runs and workflows.

Interacting with workflow runs
gh run list:      List recent workflow runs
gh run view:     View details for a workflow run or one of its jobs
gh run watch:    Watch a workflow run while it executes
gh run rerun:    Rerun a failed workflow run
gh run download: Download artifacts generated by runs

To see more help, run `gh help run <subcommand>`
```



```
Interacting with workflow files
gh workflow list:  List workflow files in your repository
gh workflow view:  View details for a workflow file
gh workflow enable: Enable a workflow file
gh workflow disable: Disable a workflow file
gh workflow run:   Trigger a workflow_dispatch run for a workflow file

To see more help, run `gh help workflow <subcommand>`
```



```
Interacting with the GitHub Actions cache
gh cache list:    List all the caches saved in GitHub Actions for a repository
gh cache delete:  Delete one or all saved caches in GitHub Actions for a repository

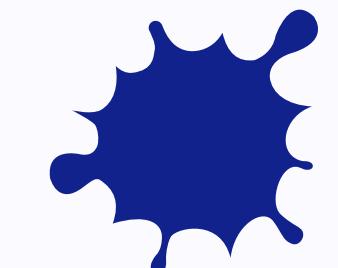
To see more help, run `gh help cache <subcommand>`
```

GitHub CLI pozwala na sterowanie działaniem GitHub Actions za pomocą trzech zestawów poleceń

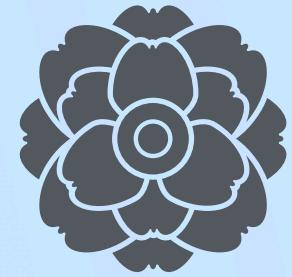
Przykład działania:

```
~/examples/L10 🐫 main [✓]
> gh workflow list
```

NAME	STATE	ID
GHAction example (END)	active	157415380



Do danego łańcucha CI (workflow) można odwoływać się poprzez jego numer



Github Actions + gh CLI - cz. II

```
apple ~ /Labs/lab9 🏙 main [✓]
> gh help workflow
List, view, and run workflows in GitHub Actions.

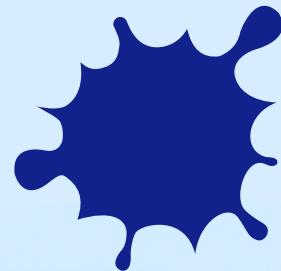
USAGE
  gh workflow <command> [flags]

AVAILABLE COMMANDS
  disable:      Disable a workflow
  enable:       Enable a workflow
  list:         List workflows
  run:          Run a workflow by creating a workflow_dispatch event
  view:         View the summary of a workflow
```

Polecenie `gh workflow run` uruchamia działanie łańcucha CI WYŁĄCZNIE jeśli zdeklarowany jest wyzwalacz (event) w postaci `workflow_dispatch`

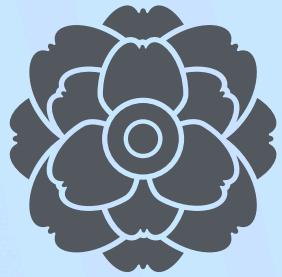
W przykładowym łańcuchu CI dla Github Actions zdeklarowane są dwa wyzwalacze:

- przesłanie nowego tag-u do repo git (gałąź main): uruchamia tagowanie obrazu wg. Semver
- Ręczne uruchomienie łańcucha CI: uruchamia tagowania bazujące na sha ostatniego commit-a (na gałęzi main)



Pełna dokumentacja metod wyzwalania workflows:

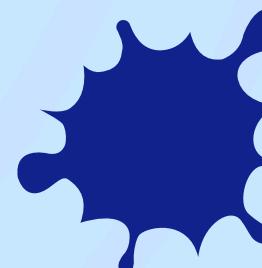
<https://docs.github.com/en/actions/using-workflows/triggering-a-workflow>



Github Actions - użycie workflow_dispatch - cz. I

1

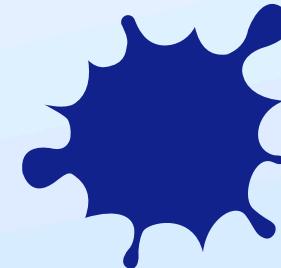
Uruchomienie
łańcucha CI



```
❯ gh workflow run  
? Select a workflow GHAction example (gha_example.yml)  
✓ Created workflow_dispatch event for gha_example.yml at main  
  
To see runs for this workflow, try: gh run list --workflow=gha_example.yml
```

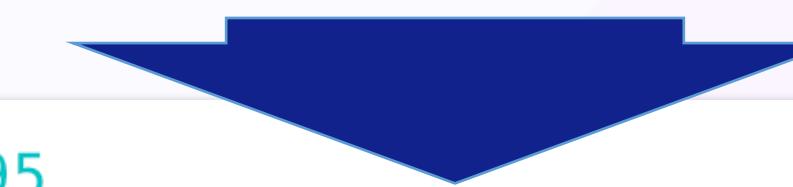
2

Sprawdzenie statusu wykonania łańcucha CI

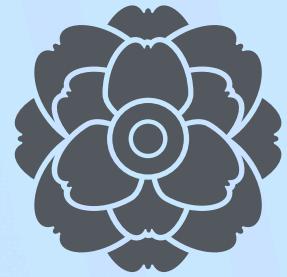


W trakcie realizacji danego
workflow można
obserwować go za pomocą
polecenia `gh run watch`

```
❯ gh run view  
? Select a workflow run * GHAction example, GHAction example [main] 10s ago
```



```
* main GHAction example · 14602418395  
Triggered via workflow_dispatch less than a minute ago  
  
JOBS  
* Build, tag and push Docker image to DockerHub (ID 40963517405)  
  
For more information about the job, try: gh run view --job=40963517405  
View this run on GitHub: https://github.com/SenatorP51/lab10/actions/runs/14602418395  
  
(END)
```



Github Actions - użycie workflow_dispatch - cz. II

3

Szczegóły realizacji danego łańcucha CI



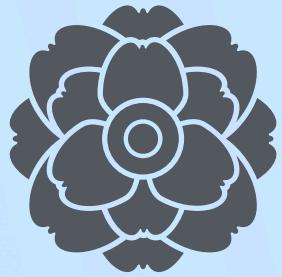
```
apple ~ /examples/L10 🐫 main [✓]
> gh run view --job=40963517405
```

```
✓ main GHAction example · 14602418395
triggered via workflow_dispatch less than a minute ago

✓ Build, tag and push Docker image to DockerHub in 30s (ID 40963517405)
  ✓ Set up job
  ✓ Check out the source_repo
  ✓ Docker metadata definitions
  ✓ QEMU set-up
  ✓ Buildx set-up
  ✓ Login to DockerHub
  ✓ Build and push Docker image
  ✓ Post Build and push Docker image
  ✓ Post Login to DockerHub
  ✓ Post Buildx set-up
  ✓ Post QEMU set-up
  ✓ Post Check out the source_repo
  ✓ Complete job
```

To see the full job log, try: `gh run view --log --job=40963517405`
View this run on GitHub: <https://github.com/SenatorP51/lab10/actions/runs/14602418395>

(END)



PAwChO – Laboratorium 10

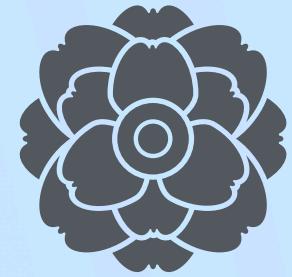
Github Actions - użycie workflow_dispatch - cz. III

Sprawdzenie poprawności wykonanych działań - GitHub:

The screenshot shows two GitHub repository pages. The top page displays the 'Actions' tab for the repository 'SenatorP51 / lab10'. It lists a workflow named 'GHAction example' with one run. A large blue arrow points from this screen to the bottom one. The bottom page shows the details of the first workflow run for 'GHAction example #1'. It includes sections for 'Summary', 'Jobs', and 'Run details'. A red box highlights the 'Build, tag and push Docker ima...' job in the 'Jobs' section.

**Polecenia z grupy `gh cache` ...
pozwalają na operacje na danych
cache TYLKO jeśli te dane są
przechowywane na repozytorium
`ghcr.io`**

**„Kliknięcie” otwiera okno
szczegółów realizacji workflow**



PAwChO – Laboratorium 10

Github Actions - użycie workflow_dispatch - cz. IV

Sprawdzenie poprawności wykonanych działań - DockerHub:

„Kliknięcie” na zakładkę tag pozwala potwierdzić, że obraz wspiera dwie platformy sprzętowe

Detailed description: The screenshot shows the Docker Hub interface. On the left, the sidebar includes 'Explore', 'My Hub' (selected), 'Repositories' (highlighted in grey), 'Settings', 'Default privacy', 'Notifications', 'Billing', 'Usage', 'Pulls', and 'Storage'. The main area shows the 'spg51' user profile with a dropdown menu. Below it, the 'spg51/example' repository card is displayed, showing it was last pushed 27 minutes ago and has a size of 9.9 MB. The card includes links for 'GHActions example', 'Add a category', and tabs for 'General' (selected) and 'Tags'. A blue arrow points from the user dropdown to the repository card. A red arrow points from the 'Tags' tab to the 'cache' tag entry in the table below. A large blue arrow points from the repository card to a detailed view of the 'cache' tag.

Tag	OS	Type	Pulled	Pushed
cache	---	Image	less than 1 day	27 minutes
sha-e6dd510		Image	less than 1 day	27 minutes

The detailed view of the 'cache' tag shows the following information:

TAG
cache

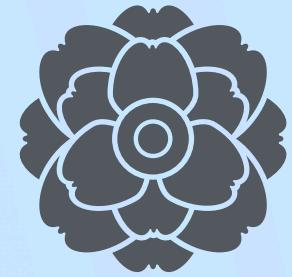
Last pushed 30 minutes by [spg51](#)

TAG
sha-e6dd510

Last pushed 30 minutes by [spg51](#)

Digest
6e20f5303665
eadf96768ec2

OS/ARCH
linux/amd64
linux/arm64



Github Actions - użycie git tags - cz. I

1

Utworzenie nowego tag-a na gałęzi main na lokalnym repozytorium git

2

Przesłanie tag-a na repo na Github (synchronizacja)

```
~/examples/L10 $ main [✓]
> git tag -a "v1.0.1" -m "test build v1.0.1"
```

```
~/examples/L10 $ main [✓]
> git push origin tag v1.0.1
Wymienianie obiektów: 1, gotowe.
Zliczanie obiektów: 100% (1/1), gotowe.
Zapisywanie obiektów: 100% (1/1), 183 bajty | 183.00 KiB/s, gotowe.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:SenatorP51/lab10.git
 * [new tag]           v1.0.1 -> v1.0.1
```

Triggered via push less than a minute ago

JOB

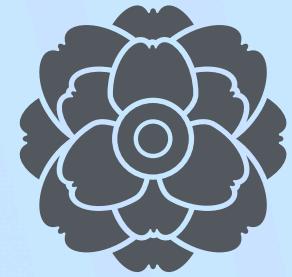
- ✓ Build, tag and push Docker image to DockerHub in 32s (ID 40965715788)
 - ✓ Set up job
 - ✓ Check out the source_repo
 - ✓ Docker metadata definitions
 - ✓ QEMU set-up
 - ✓ Buildx set-up
 - ✓ Login to DockerHub
 - ✓ Build and push Docker image
 - ✓ Post Build and push Docker image
 - ✓ Post Login to DockerHub
 - ✓ Post Buildx set-up
 - ✓ Post QEMU set-up
 - ✓ Post Check out the source_repo
 - ✓ Complete job

✓ Run GHAction example (14603088708) completed with 'success'

3

Obserwacja „na żywo” działań zdefiniowanych w ramach łańcucha CI

```
~/examples/L10 $ main [✓]
> gh run watch
? Select a workflow run * Inicjalizacja repo lab10, GHAction example [v1.0.1] 17s ago
✓ v1.0.1 GHAction example · 14603088708
```



PAwChO – Laboratorium 10

Github Actions - użycie git tags - cz. II

Sprawdzenie poprawności wykonanych działań:



```
apple ~ /examples/L10 🌐 main [✓]
> gh run view
? Select a workflow run ✓ Inicjalizacja repo lab10, GHAction example [v1.0.1] 8m46s ago
```

```
✓ v1.0.1 GHAction example · 14603088708
Triggered via push about 8 minutes ago

JOBS
✓ Build, tag and push Docker image to DockerHub in 32s (ID 40965715788)

For more information about the job, try: gh run view --job=40965715788
View this run on GitHub: https://github.com/SenatorP51/lab10/actions/runs/14603088708

(END)
```

spg51/example

Last pushed 13 minutes ago · Repository size: 9.9 MB

GHActions example

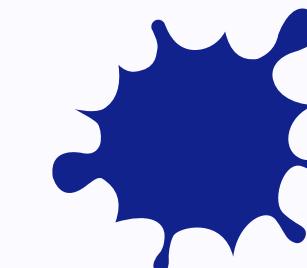
Add a category

General Tags Image Management BETA Collaborators Webhooks Settings

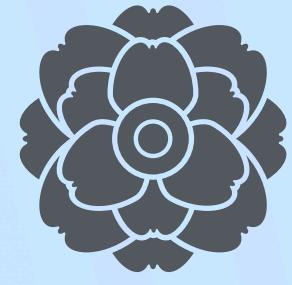
Tags

This repository contains 3 tag(s).

Tag	OS	Type	Pulled	Pushed
cache	---	Image	less than 1 day	13 minutes
sha-e6dd510		Image	less than 1 day	13 minutes
1.0.1		Image	less than 1 day	13 minutes



Poprzez sterowanie doborem wyzwalaczy można tworzyć obrazy o tag-owaniu wymaganym przez ewentualne, kolejne etapy działań w ramach łańcucha (np, testy czy CD)

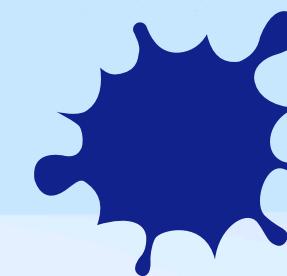


PAwChO – Laboratorium 10

Omówienie zadania nr 2



Treść zadania nr 2 jest dostępna na moodle w dedykowanym katalogu. Realizacja tego zadania opiera się na przykładach przedstawionych na tym laboratorium oraz na informacjach przedstawionych na wykładzie 8. Termin realizacji zadania zostanie podany przez prowadzącego.



Z laboratorium 10 nie jest wymagane wykonanie sprawozdania.