Dokumentation M120

By Nando Sterki + Jori Villiger

# Day 1

Creating 'Owner' role assignment under scope '/subscriptions/965aa8a5-9a39-4f68-8908-06e3c4d82d85'

The output includes credentials that you must protect. Be sure that you do not include these credentials in your code or check the credentials into your source control. For more information, see https://aka.ms/azadsp-cli

#Username, password, someStringThatIsNeeded

#Username

eac396b0-c31e-4c92-af4c-5e2c19d132ba

#Tenant

02b7e332-b310-470b-8eef-729bfcf99667

#Password

Cr.8Q~NCijtNw5EpSwa\_SPy7IQ0~uNxESXOXjdl6

Connect to server (Task 1)

az mysql flexible-server connect -n studle1server -u sterki -interactive

Create App (Task 2)

az spring app create \

    --resource-group NS\_Studle\_DB \

    --service studle2 \

    --name studle1app \

    --runtime-version Java\_17 \

    --assign-endpoint true

https://studle2-studletheapp.azuremicroservices.io

Deploy App (last task)

az spring app deploy \

    --resource-group NS\_Studle\_DB \

    --service studle2 \

    --name studle1app\

    --artifact-path studle-app.jar

Get set up on server (Task 3)

    az spring connection create mysql-flexible \

    -g NS\_Studle\_DB \

    --tg NS\_Studle\_DB \

    --connection studle1connection \

    --service studle2 \

    --app studle1app \

    --server studle1server \

    --database ns-studle \

    --client-type springBoot \

    --secret name=sterki secret=Judihui123

Set everything up (Task 4)

az mysql flexible-server parameter set \

    --resource-group NS\_Studle\_DB \

    --server-name ns-studle \

    --name require\_secure\_transport \

    --value OFF \

Ame

Delete all (when youre done)

az spring app delete --name studle1app \

                     -g  NS\_Studle\_DB \

                     -s studle2

az spring delete --name studle2 -g NS\_Studle\_DB

Logs in einer Pipeline:

CI/CD > Jobs > Status:passed

.gitlab-ci.yml

default:

image: alpine:3

job1:

stage: build

script:

- echo "Executed on all branches"

job2:

stage: build

script:

- echo "Executed only on the production branch"

rules:

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /main/

when: on\_success

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /test1/

when: never

job3:

stage: build

script:

- echo "Executed on the branches main and staging"

rules:

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /main/

when: on\_success

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /staging/

when: on\_success

- if: $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == /test1/

when: never

Tipps & Tricks:

* Keine Tabs erlaubt => 2 Leerschläge == 1 Tab
* $CI\_MERGE\_REQUEST\_SOURCE\_BRANCH\_NAME == aktueller Branchname
* When-Attribute :
  + on\_success

default, wenn alle Jobs erfolgreich durchlaufen

* + manual

nur wenn Job manuell getriggert wird

* + always

immer, Status vom Job egal (auch wenn er failed)

* + on\_failure

nur wenn min. ein Job failt

* + delayed

verzögert um eine angegebene Zeit

* + never

niemals

* equals (==)
* not equals (!=)
* and (&&)
* or (||)

Das funktioniert nur mit absolut leeren Reps:

cd existing\_folder

git init --initial-branch=main

git remote add origin PFAD\_ZU\_DEINEM\_GITLAB\_REPO

git add .

git commit -m "Initial commit"

git push -u origin main

für möglichst fehlerfreies Ergebnis:

die remote branch sollte im GUI noch nicht existieren (nur bei first commits)

