STEP1 : Take a simple code of Hello World in Python:

```
from time import sleep,
# the program will print hello world
# every 1 second foever
while True:
print("Hello, World")
sleep(1)
```

Step2: Create a Docker File of code

```
FROM python:3,
RUN mkdir WORK_REPO
RUN cd WORK_REPO
WORKDIR /WORK_REPO
ADD hello_world.py .
CMD ["python", "-u", "hello_world.py"]
```

STEP 3: Build an image from docker file

```
docker build -t hello_world:v1 .
```

STEP4: Check Image and Run the container

docker images

docker run -d hello_world:v1

docker push (Docker HUB)

STEP5: Install argocd image updater

The most straightforward way to run the image updater is to install it as a Kubernetes workload into the namespace where Argo CD is running

kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj-labs/argocd-image-updater/stable/manifests/install.yaml

Must check all pods are up and running in argocd namespace.

Create git credential secret file for your argocd application. Behind the scene ArgoCD uses these secret to sync your repository.

```
kubectl --namespace argocd create secret generic git-creds --
from-literal=username=<username> --from-
literal=password=<token>
```

STEP 6: Create Application

```
apiVersion: argoproj.io/v1alpha
kind: Application
metadata:
  name: <application name>
  namespace: argocd
  annotations:
    argocd-image-updater.argoproj.io/image-list: myalias= <docker
hub repo)
    argocd-image-updater.argoproj.io/write-back-method:
git:secret:argocd/git-creds
    argocd-image-updater.argoproj.io/git-branch: main
    argocd-image-updater.argoproj.io/myalias.force-update: "true"
spec:
  project: default
  source:
    repoURL:    <repo-name>
    targetRevision: HEAD
    path: dev
  destination:
    server: https://kubernetes.default.svc
    namespace: <application namespace>
  syncPolicy:
    syncOptions:
    - CreateNamespace=true
    automated:
      selfHeal: true
      prune: true
```

```
kubectl apply -f application.yaml
```

STEP6: TESTING

From Step1 change some code and rebuild the image and again push the image to the docker hub.

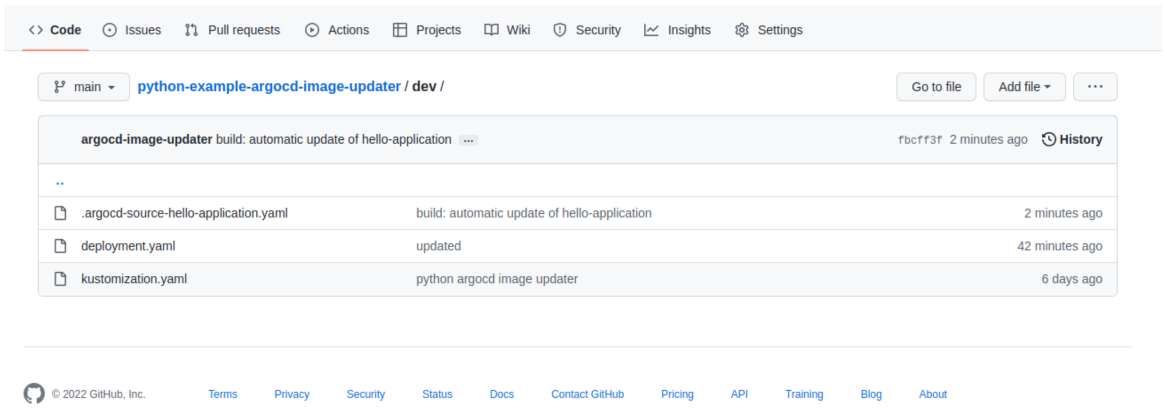
```
from time import sleep,
# the program will print hello world
# every 1 second foever
while True:
print("Hello, World 2 ")
sleep(1)
```

```
docker build -t hello_world:v2 .

docker run -d hello_world:v2

docker push (Docker HUB)
```

For checking image updater logs :

```
kubectl --namespace argocd logs --selector
app.kubernetes.io/name=argocd-image-updater --follow
```

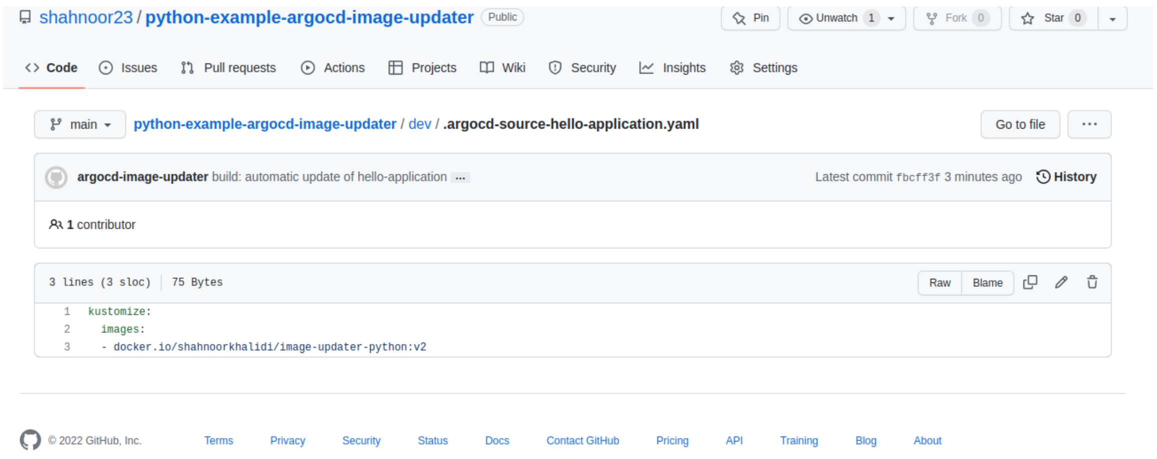
As soon as you will pushed your image to your container registry **it will automatically fetch it's latest tags** and depending your conditions defined in application and **it's creates one extra file on your repository at same path where manifest are located.**



python-example-argocd-image-updater / dev /

argocd-image-updater build: automatic update of hello-application ... fbcff3f 2 minutes ago

..		
.argocd-source-hello-application.yaml	build: automatic update of hello-application	2 minutes ago
deployment.yaml	updated	42 minutes ago
kustomization.yaml	python argocd image updater	6 days ago


shahnoor23 / python-example-argocd-image-updater Public

python-example-argocd-image-updater / dev / .argocd-source-hello-application.yaml

argocd-image-updater build: automatic update of hello-application ...

Latest commit fbcff3f 3 minutes ago

1 contributor

3 lines (3 sloc) | 75 Bytes

```yaml
kustomize:
  images:
  - docker.io/shahnoorkhalidi/image-updater-python:v2
```