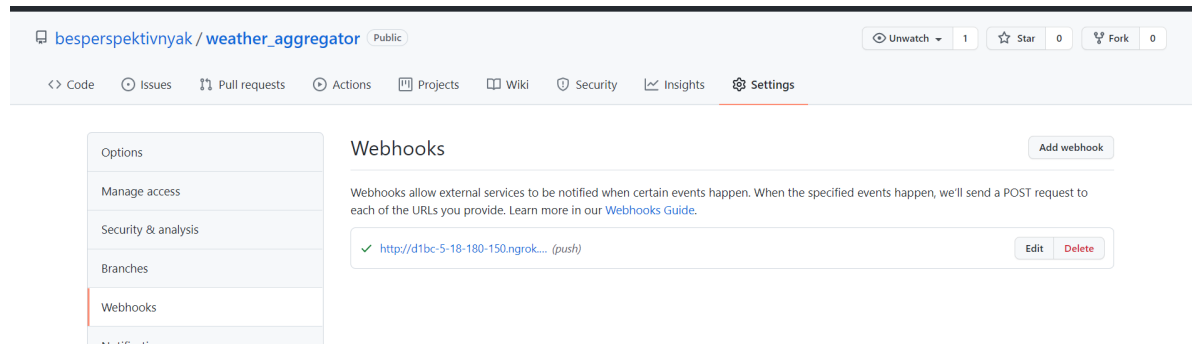


Manage pipeline

GitHub

Add webhook to our project. As payload URL use generated by ngrok host.



We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *

http://d1bc-5-18-180-150.ngrok.io/github-webhook/

Content type

application/json

Secret

Which events would you like to trigger this webhook?

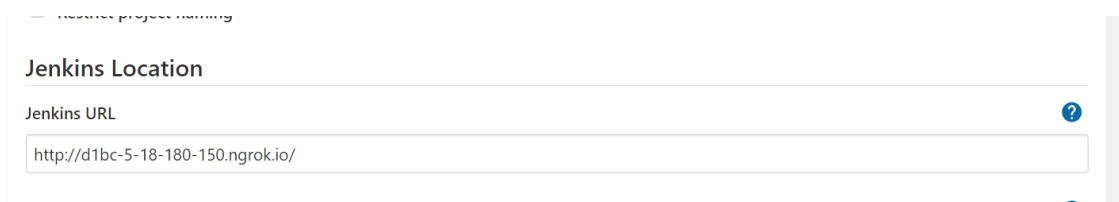
- ☒ Just the push event.
- ☐ Send me **everything**.
- ☐ Let me select individual events.

☒ Active

We will deliver event details when this hook is triggered.

Jenkins

In Jenkins system configuration change Jenkins URL on ngrok host on previous step.



Then create freestyle project with following settings.

There is the link on your GitHub project.

[Plain text] Предпросмотр

☒ GitHub project

Project url ?

Расширенные...

☐ This build requires lockable resources

☐ Throttle builds ?

☐ Удалять устаревшие сборки ?

☐ Это - параметризованная сборка ?

☐ Приостановить сборки ?

☐ Разрешить параллельный запуск задачи ?

Расширенные...

The same URL

Управление исходным кодом

☐ Нет

☒ Git ?

Repositories ?

Repository URL ?

Credentials ?

Add

Расширенные...

Add Repository

Сохранить Применить ?

and name of branch

Branches to build ?

Branch Specifier (blank for 'any') X ?

Add Branch

Просмотрщик репозитория ?

Additional Behaviours

Добавить

Our Jenkins project will be triggered by every GitHub push.

Триггеры сборки

- ☐ Build after other projects are built
- ☒ GitHub hook trigger for GITScm polling
- ☐ Запускать периодически
- ☐ Опрашивать SCM об изменениях

Среда сборки

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s)
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☐ Inspect build log for published Gradle build scans
- ☐ With Ant

Сборка

The last step: rebuild docker image and push it on DockerHub.

Сборка

Выполнить команду Windows

Команда

```
docker build -t besperspektivnyak/weather . Example of my image
docker login -u Your DockerHub login -p Your password
docker image push besperspektivnyak/weather
```

See [the list of available environment variables](#)


Расширенные...

Добавить шаг сборки ▾

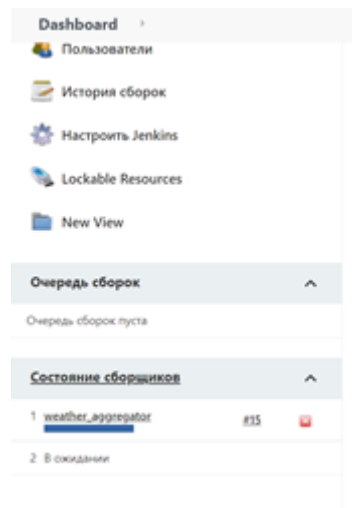
After these steps your Docker image will update after every commit pushed on GotHub.

Example

I've did test commit in repository of my project.

	besperspektivnyak test commit	3936533 2 days ago	🕒 23 commits
📁	weather_aggregator	delete page at /	2 days ago
📄	Dockerfile	better version of Dockerfile	3 days ago
📄	README.md	test commit	2 days ago
📄	requirements.txt	move requirements.txt	17 days ago
📄	url.py	add api key to env variables	7 days ago


In Jenkins have started build.



Build ended with success (see console output below).

Then check out DockerHub and see new push.



besperspektivnyak / weather

weather_aggregator 


 Last pushed: 2 days ago

Tags and Scans

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
 latest		a few seconds ago	2 days ago

[See all](#)

 VULNERABILITY SCANNING - DISABLED
[Enable](#)

That's all!

Ngrok

To get your own host I've used Ngrok.

Firstly, sign up on [Ngrok](#). Then follow the instruction to connect your account. As final step you need to use command with port on which Jenkins is started.

```
ngrok.exe http <your port>
```

You'll see the following picture in your terminal. Copy necessary URL and paste it in GitHub Webhook and as Jenkins URL (see the beginning).

```
Session Status      online
Account             Elizaveta (Plan: Free)
Version             2.3.40
Region              United States (us)
Web Interface        http://127.0.0.1:4041
Forwarding           http://d1bc-5-18-180-150.ngrok.io -> http://localhost:8080
Forwarding           https://d1bc-5-18-180-150.ngrok.io -> http://localhost:8080
```



Назад к проекту



Статус



Изменения



Вывод консоли



Просмотреть как неформатированный текст



Редактировать информацию сборки



Delete build '#15'



Лог опроса



Git Build Data



Предыдущая сборка



Вывод на консоль

Started by GitHub push by besperspektivnyak

Running as SYSTEM

Building in workspace

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\weather_aggregator

The recommended git tool is: NONE

No credentials specified

> git.exe rev-parse --resolve-git-dir

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\weather_aggregator\.git # timeout=10

Fetching changes from the remote Git repository

> git.exe config remote.origin.url

https://github.com/besperspektivnyak/weather_aggregator.git # timeout=10

Fetching upstream changes from https://github.com/besperspektivnyak/weather_aggregator.git

> git.exe --version # timeout=10

> git --version # 'git version 2.25.0.windows.1'

> git.exe fetch --tags --force --progress --

https://github.com/besperspektivnyak/weather_aggregator.git

+refs/heads/*:refs/remotes/origin/* # timeout=10

```
> git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision 39365331427f47d725b98ec65a47b13d27870d21 (refs/remotes/origin/main)
> git.exe config core.sparsecheckout # timeout=10
```

Dashboard ▶ weather_aggregator ▶ #15

Commit message: "test commit"

```
> git.exe rev-list --no-walk 8335788008f2415c6471f7601c7c4c5ccf89aae1 # timeout=10
[weather_aggregator] $ cmd /c call C:\WINDOWS\TEMP\jenkins6628549818931049718.bat
```

```
C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\weather_aggr
egator>docker build -t besperspektivnyak/weather .
```

```
#1 [internal] load build definition from Dockerfile
```

```
#1 sha256:1cbf3dacd66ff62f52456a28b4c101cbb99153cc64e11fc759b8930d08f9fe70
```

```
#1 ...
```

```
#2 [internal] load .dockerignore
```

```
#2 sha256:c676a816cecf5956ef1a255a2fb8b3cb1a0b336b03c2bc4a538ab41e6017fe58
```

```
#2 transferring context: 2B 0.0s done
```

```
#2 DONE 0.4s
```

```
#1 [internal] load build definition from Dockerfile
```

```
#1 sha256:1cbf3dacd66ff62f52456a28b4c101cbb99153cc64e11fc759b8930d08f9fe70
```

```
#1 transferring dockerfile: 32B 0.1s done
```

```
#1 DONE 0.6s
```

```
#3 [internal] load metadata for docker.io/library/python:3.8.0
```

```
#3 sha256:0fccb25f9bdaa37f1acc828d268bb0353849363fe7a76d73721e528c9bd3b29c
```

```
#3 ...
```

```
#4 [auth] library/python:pull token for registry-1.docker.io
```

```
#4 sha256:ba5f0b9b44583e04460459d5765aeea1c44fe24897c87abe8323fd6f56245605
```

```
#4 DONE 0.0s
```

```
#3 [internal] load metadata for docker.io/library/python:3.8.0
```

```
#3 sha256:0fccb25f9bdaa37f1acc828d268bb0353849363fe7a76d73721e528c9bd3b29c
```

```
#3 DONE 9.5s
```

```
#5 [1/6] FROM
```

```
docker.io/library/python:3.8.0@sha256:adb48bf76e44cd7d74607db157bba756368af42196aaea945e2114d9
57cb5558
```

```
#5 sha256:1f6050661818bb47af045e605c6bb45c15abec6163a58eff602fcea9080a600f
```

```
#5 DONE 0.0s
```

```
#8 [internal] load build context
```

```
#8 sha256:4d34a0a43849ac530bf396aacc18b882813d07812817934fa45915db84691ac3
```

```
#8 transferring context: 24B 0.0s
```

```
#8 transferring context: 17.76kB 0.2s done
```

```
#8 DONE 0.3s
```

```
#9 [4/6] COPY requirements.txt .
```

```
#9 sha256:fac5ea613244d24e40a2d64b3e91c08d61e6c681c28b02aad7493c575d99ab79
```

```
#9 CACHED
```

```
#6 [2/6] WORKDIR /app
```

```
#6 sha256:e43c7544a5aa53cbb2ccdc3a9b7b33fa5a782a6aaee6ccd6c2dff7e95f871f3
```

#6 CACHED

#7 [3/6] RUN apt-get update -y && apt-get install -y python3-pip

Dashboard ▶ **weather_aggregator** ▶ **#15**

#10 [5/6] RUN pip install --no-cache-dir -r requirements.txt

#10 sha256:65f149c36ab69a3e82219ee72096cfde47b67bc991a20ff7df7de101ed51241d

#10 CACHED

#11 [6/6] COPY . .

#11 sha256:9979c99fea1dd953b4c80758f1c440ae6327b8cb6a4be9855786918e00ec1827

#11 DONE 0.4s

#12 exporting to image

#12 sha256:e8c613e07b0b7ff33893b694f7759a10d42e180f2b4dc349fb57dc6b71dcab00

#12 exporting layers

#12 exporting layers 0.3s done

#12 writing image sha256:1e103b26e536f59ec97ed0982e97a0d45ac9597f59ac99894ef4ddd3c3fecfcb 0.0s done

#12 naming to docker.io/besperspektivnyak/weather 0.0s done

#12 DONE 0.5s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\weather_aggregator>docker login -u besperspektivnyak -p dom3490041

WARNING! Using --password via the CLI is insecure. Use --password-stdin.

Login Succeeded

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\weather_aggregator>docker image push besperspektivnyak/weather

Using default tag: latest

The push refers to repository [docker.io/besperspektivnyak/weather]

0780c5fdef3a: Preparing

9c71521b23ae: Preparing

db20e20e57ba: Preparing

635010e08df0: Preparing

b5e8827c38c0: Preparing

00947a3aa859: Preparing

7290ddeeb6e8: Preparing

d3bfe2faf397: Preparing

cecea5b3282e: Preparing

9437609235f0: Preparing

bee1c15bf7e8: Preparing

00947a3aa859: Waiting

7290ddeeb6e8: Waiting

d3bfe2faf397: Waiting

cecea5b3282e: Waiting

9437609235f0: Waiting

bee1c15bf7e8: Waiting

423d63eb4a27: Preparing

7f9bf938b053: Preparing

f2b4f0674ba3: Preparing

423d63eb4a27: Waiting
7f9bf938b053: Waiting
f2b4f0674ba3: Waiting

Dashboard ▶ **weather_aggregator** ▶ **#15**

db20e20e57ba: Layer already exists
635010e08df0: Layer already exists
00947a3aa859: Layer already exists
d3bfe2faf397: Layer already exists
cecea5b3282e: Layer already exists
7290ddeeb6e8: Layer already exists
423d63eb4a27: Layer already exists
bee1c15bf7e8: Layer already exists
9437609235f0: Layer already exists
7f9bf938b053: Layer already exists
0780c5fdef3a: Pushed
f2b4f0674ba3: Layer already exists
latest: digest: sha256:d51f6733a574d0ce22808eaa3176fc874c34f661af778c8c537c1516f694934c size:
3264

C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins\workspace\weather_aggr
egator>exit 0
Finished: SUCCESS