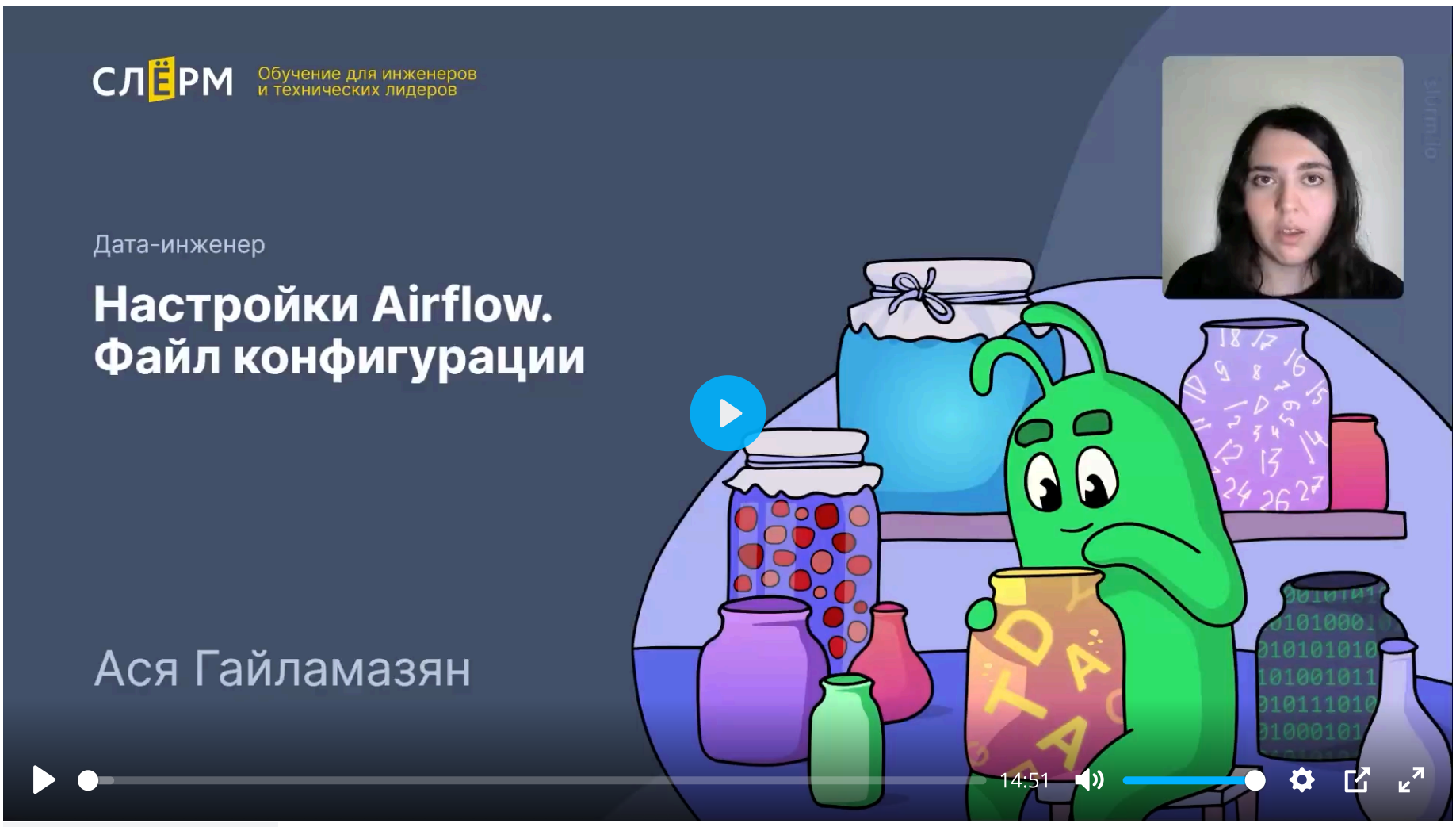
8.5 Настройка конфигурации оркестратора и мониторинг



Презентация к уроку 8.5.1

Текстовая расшифровка видео:

НАСТРОЙКИ AIRFLOW. ФАЙЛ КОНФИГУРАЦИИ

План:

- Дефолтный конфиг (Default config);
- Таймзоны;
- Pools;
- Переменные в Airflow. Задание переменных;
- Считывание кода DAG;
- Источники переменных;
- Разбор примера.

Дефолтный конфиг (Default config)

Файл конфигурации Airflow помогает настраивать инстанс. Эти настройки очень гибкие. Airflow отличается от многих оркестраторов тем, что в нем можно настраивать практически все.

Мы можем скопировать файл конфигурации, **туда не будут вноситься изменения**:

```
sudo docker cp airflow_airflow-scheduler_1:/opt/airflow/airflow.cfg .
```

С документацией по параметрам вы можете ознакомиться, перейдя по ссылке: https://airflow.apache.org/docs/apache-airflow/stable/configurations-ref.html.

Таймзоны

Один из важных параметров – Таймзона. По умолчанию **Таймзона=utc**. Все таски будут выполняться на три часа позднее, если вы находитесь в utc+3.

Вы можете задавать таймзону либо в файле конфигурации, либо в DAG (в параметре **start_date=pendulum.datetime(конкретная_таймзона)**:

```
default_timezone = utc

import pendulum
…
start_date = pendulum.datetime(2023, 1, 1, tz = “UTC”)
```

Pools

```
aggregate_db_message_job=BashOperator(
task_id='task_1',
pool='non_default_pool',
bash_command="echo 1",
dag=dag,
)
```

Механизм Pools позволяет ограничивать одновременно выполняющиеся процессы – таски.

По умолчанию Pools выставлен на **128**, то есть на машине может выполняться 128 одновременных процессов тасков. Если 128 много для сервера, можно в каждом DAG выставить отдельный параметр того, чему должен быть равен pool. Например, это можно сделать для очень тяжеловесной задачи, искусственно ограничивая количество Pool’ов, которая она может занять. Так, задача будет выполняться дольше по времени, но в то же время не будет занимать все мощности сервера.

Переменные в Airflow. Задание переменных

Переменные в Airflow можно задавать:

- Через **WebUI**;
- Через **CLI** командой **airflow variables set**;
- В коде **DAG** с помощью импорта **Variable**.

```
from airflow.models import Variable
my_var=Variable.set(“my_key”, “my_value”)
```

Считывание кода DAG

В **mine_file_process_interval** лежит значение «30», то есть код всех DAG парсится каждые тридцать секунд. Каждые тридцать секунд выполняется тот код, который находится вне DAGs. Не следует писать код вне тасков! Это будет замедлять программу.

Источники переменных

Последовательность наблюдения Airflow за переменными:

- Airflow variables.get;
- Backend Secret (Kubernetes, сохранение в облаке);
- AIRFLOW_ENV;
- Metastore (Если заданы переменные в файле конфигурации, они могут перетираться теми, которые указаны в

Разбор примера

Для начала посмотрим в файл конфигурации. Мы скопировали его, чтобы было проще смотреть через nano:

```
File  Edit  View  Search  Terminal  Help
asya@asya-Aspire-XC-886:~/airflow_edu$ ls
airflow.cfg  config  dags  docker-compose.yaml  logs  plugins  venv
asya@asya-Aspire-XC-886:~/airflow_edu$ nano airflow.cfg
```

Здесь находятся несколько разделов «core», находится то, что относится к WebUI и т.д. Например, директория «DAGs folder» – директория, где мы ищем DAGs:

```
File  Edit  View  Search  Terminal  Help
  GNU nano 4.8                                  airflow.cfg
[core]
# The folder where your airflow pipelines live, most likely a
# subfolder in a code repository. This path must be absolute.
dags_folder = /opt/airflow/dags

# Hostname by providing a path to a callable, which will resolve the hostname.
# The format is "package.function".
#
# For example, default value "airflow.utils.net.getfqdn" means that result from patched
# version of socket.getfqdn() - see https://github.com/python/cpython/issues/49254.
#
# No argument should be required in the function specified.
# If using IP address as hostname is preferred, use value ``airflow.utils.net.get_host_ip_address``
hostname_callable = airflow.utils.net.getfqdn

# A callable to check if a python file has airflow dags defined or not
# with argument as: `(file_path: str, zip_file: zipfile.ZipFile | None = None)`
# return True if it has dags otherwise False
# If this is not provided, Airflow uses its own heuristic rules.
might_contain_dag_callable = airflow.utils.file.might_contain_dag_via_default_heuristic

# Default timezone in case supplied date times are naive
# can be utc (default), system, or any IANA timezone string (e.g. Europe/Amsterdam)
default_timezone = utc

# The executor class that airflow should use. Choices include
# ``SequentialExecutor``, ``LocalExecutor``, ``CeleryExecutor``, ``DaskExecutor``,
# ``KubernetesExecutor``, ``CeleryKubernetesExecutor`` or the
# full import path to the class when using a custom executor.
executor = SequentialExecutor

# This defines the maximum number of task instances that can run concurrently per scheduler in
# Airflow, regardless of the worker count. Generally this value, multiplied by the number of
# schedulers in your cluster, is the maximum number of task instances with the running
# state in the metadata database.
parallelism = 32

# The maximum number of task instances allowed to run concurrently in each DAG. To calculate
# the number of tasks that is running concurrently for a DAG, add up the number of running
# tasks for all DAG runs of the DAG. This is configurable at the DAG level with ``max_active_tasks``,
# which is defaulted as ``max_active_tasks_per_dag``.
                                [ Read 1375 lines ]
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos  M-U Undo  M-A Mark Text
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line M-E Redo  M-6 Copy Text
```

Это раздел «core», где параметр – параллелизм (это то, сколько может исполняться параллельно тасков).

Активных тасков по одному DAG – 16 (по умолчанию).

Активных инстансов DAG Run для одного DAG – 16 (по умолчанию).

Параметр «examples=true» означает, что мы подгружаем примеры, которые у нас отображаются.

Здесь находится таймзона, отображающаяся в WebUI:

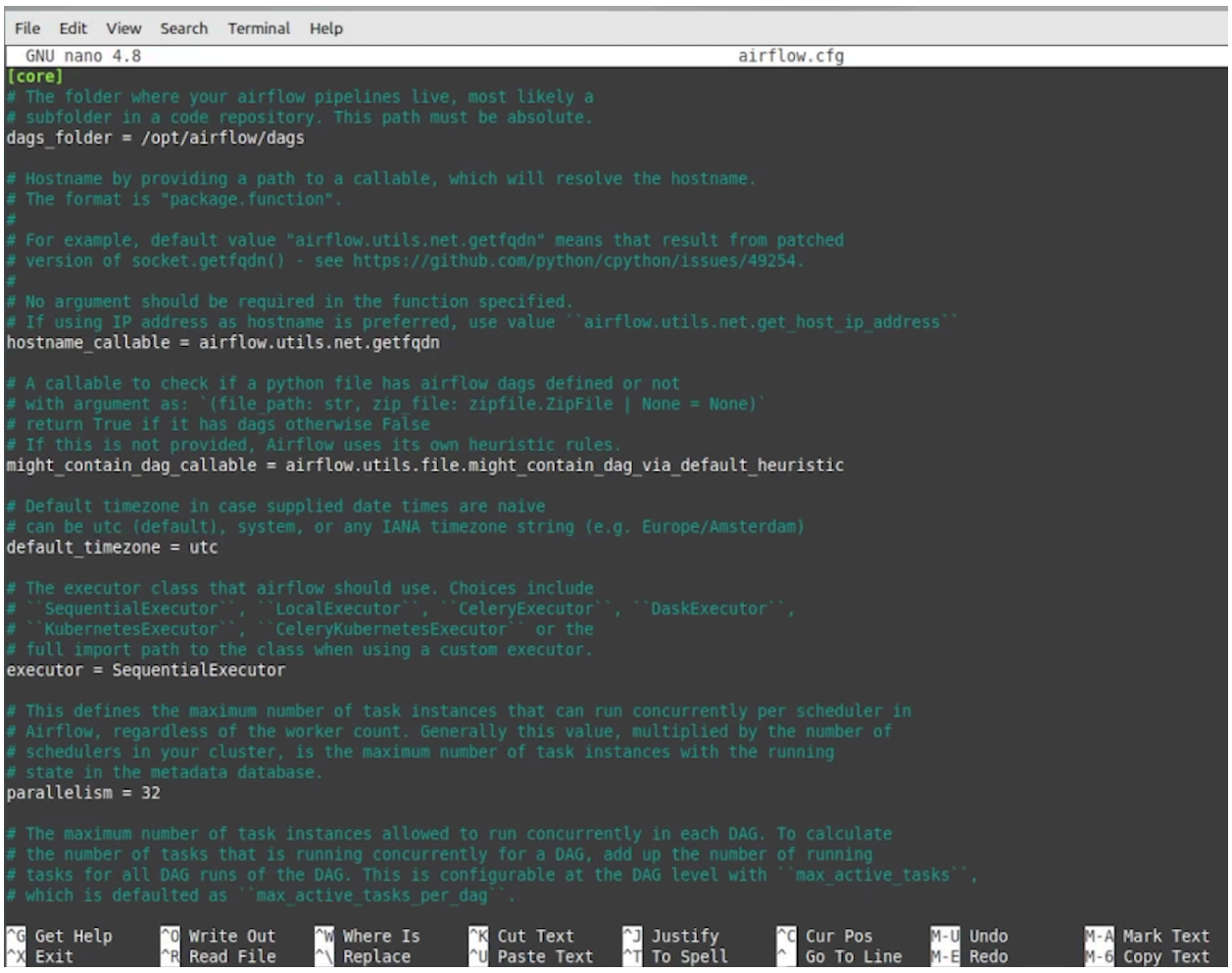
```
File  Edit  View  Search  Terminal  Help
  GNU nano 4.8                                  airflow.cfg
default_queue = default

# Is allowed to pass additional/unused arguments (args, kwargs) to the BaseOperator operator.
# If set to False, an exception will be thrown, otherwise only the console message will be displayed.
allow_illegal_arguments = False

[hive]
# Default mapreduce queue for HiveOperator tasks
default_hive_mapred_queue =

# Template for mapred_job_name in HiveOperator, supports the following named parameters
# hostname, dag_id, task_id, execution_date
# mapred_job_name_template =

[webserver]
# The base url of your website as airflow cannot guess what domain or
# cname you are using. This is used in automated emails that
# airflow sends to point links to the right web server
base_url = http://localhost:8080

# Default timezone to display all dates in the UI, can be UTC, system, or
# any IANA timezone string (e.g. Europe/Amsterdam). If left empty the
# default value of core/default_timezone will be used
# Example: default_ui_timezone = America/New_York
default_ui_timezone = UTC

# The ip specified when starting the web server
web_server_host = 0.0.0.0

# The port on which to run the web server
web_server_port = 8080

# Paths to the SSL certificate and key for the web server. When both are
# provided SSL will be enabled. This does not change the web server port.
web_server_ssl_cert =

# Paths to the SSL certificate and key for the web server. When both are
# provided SSL will be enabled. This does not change the web server port.
web_server_ssl_key =

# The type of backend used to store web session data, can be 'database' or 'securecookie'
```

Дедлайн 04 августа, 23:59 Мск

«**Timezone=UTC**» – не совсем та таймзона, отвечающая за выполнение тасков. Это та таймзона, которая отображается, когда вы смотрите в браузере. Это разные таймзоны!

Также здесь указываем url веб-сервера. Если вы хотите открыть его удаленно, то вместо localhost 8080 указывайте другой host и порт.

Default timezone – это таймзона по выполнению всех DAGs. Если вы хотите ее исправить, делать это нужно здесь:



Таймзону можно указывать в разном формате. Вместо UTC можно указывать города (в интернете можно почитать о ключевых).

В дополнительных материалах будут указаны ссылки.

Проверяя параметр Pool, видим 128:

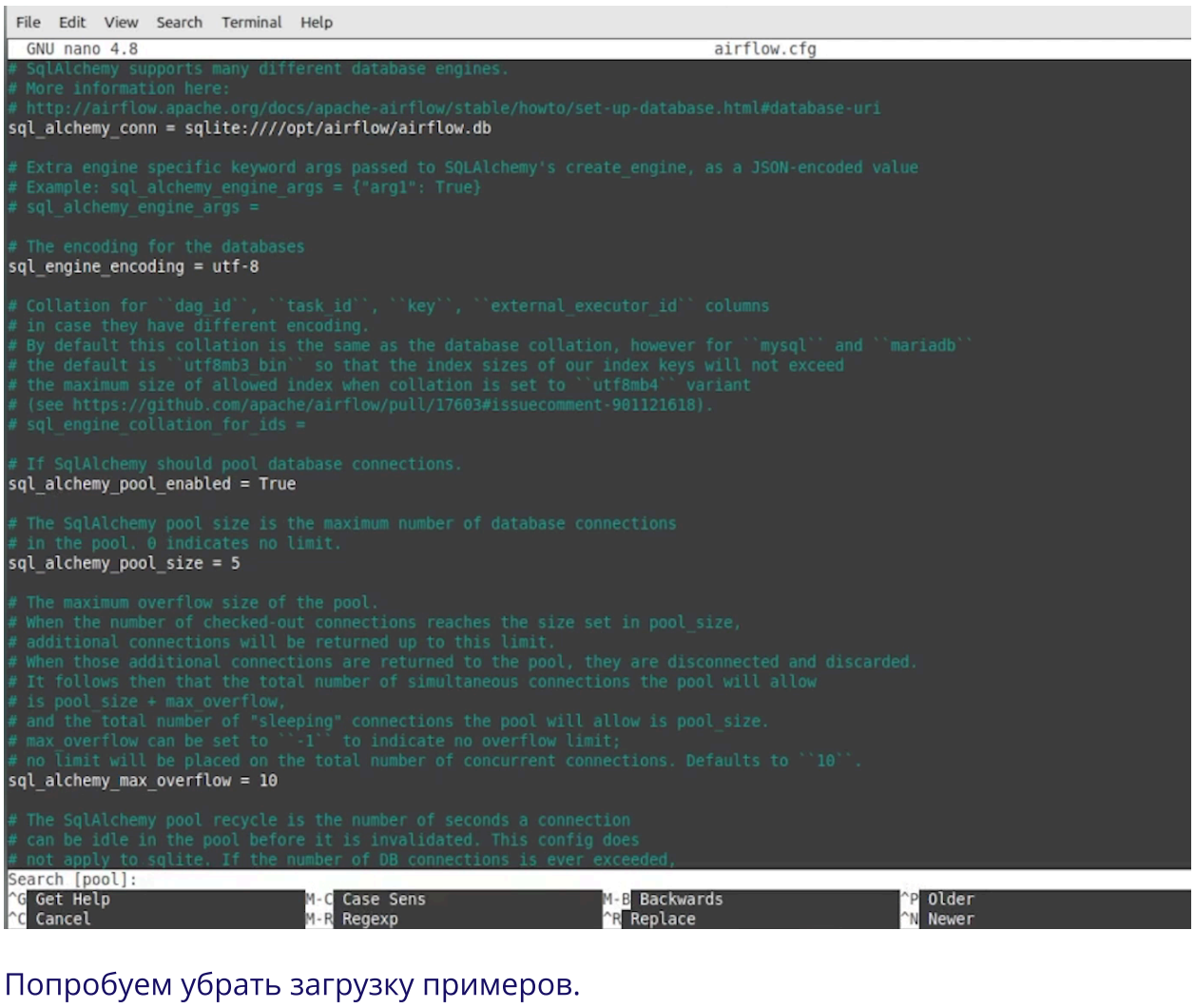
```
# Task Slot counts for ``default_pool``. This setting would not have any effect in an existing
# deployment where the ``default_pool`` is already created. For existing deployments, users can
# change the number of slots using Webserver, API or the CLI
default_pool_task_slot_count = 128
```

Вопрос: можно ли иметь несколько подключений к базе данных?

Ответ: если параметр – «true», то можно.

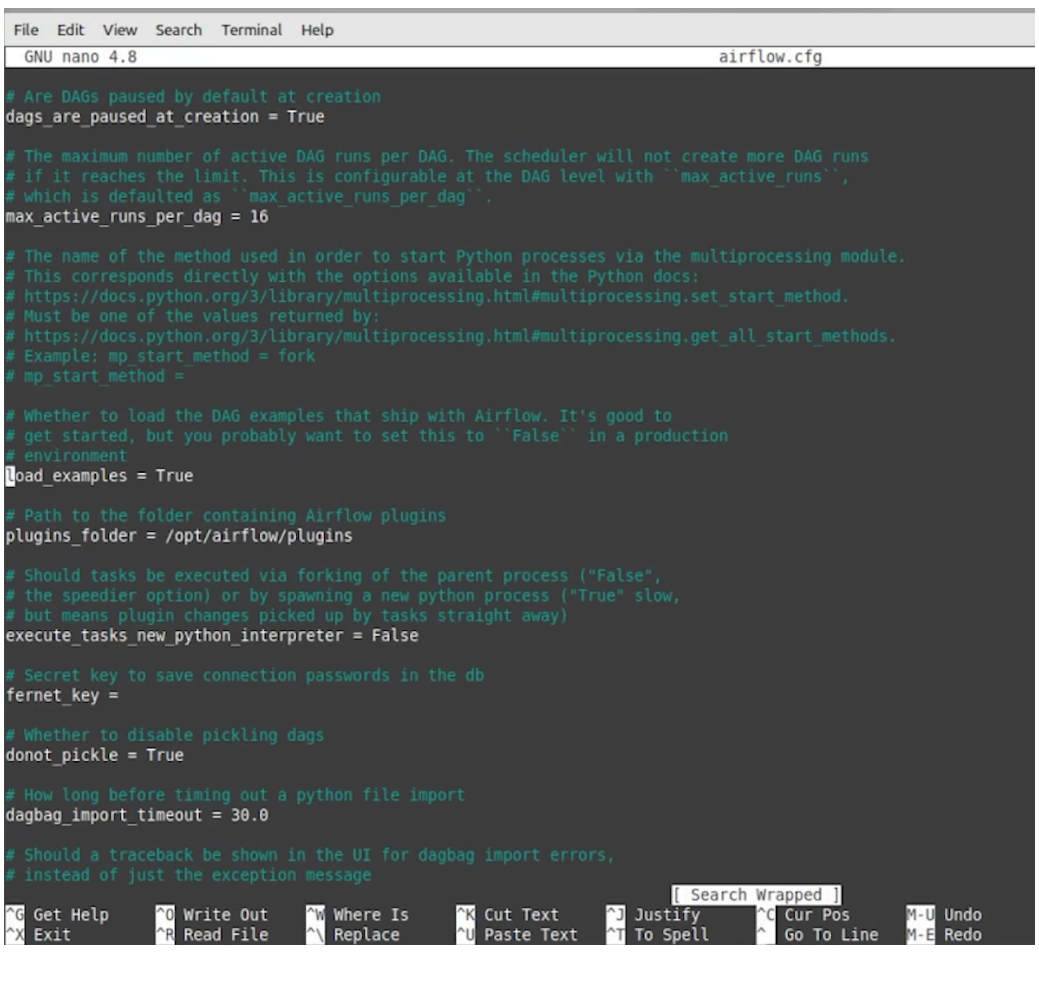
Здесь указывается максимальное количество подключений к базе данных:

Дедлайн 04 августа, 23:59 Мск

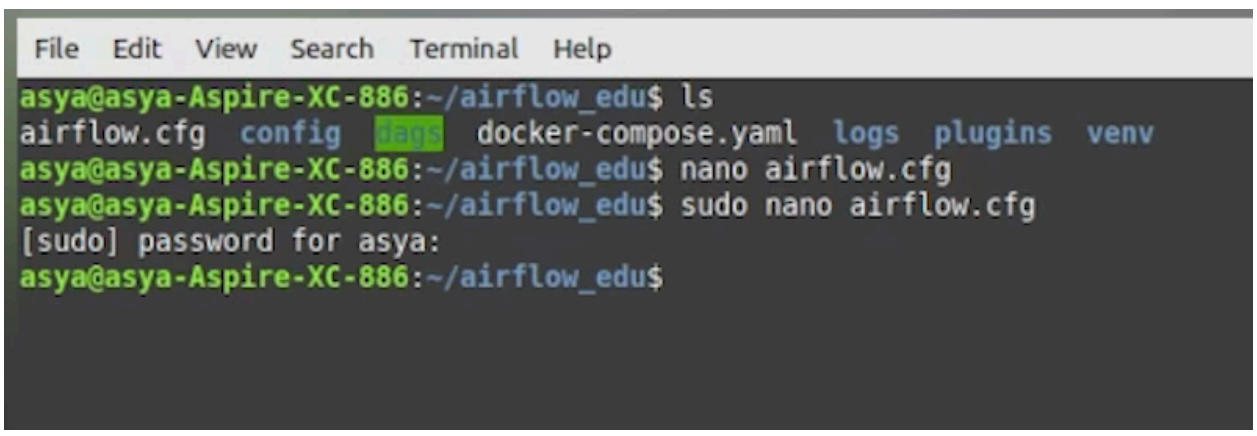


Попробуем убрать загрузку примеров.

Чтобы примеры не загружались, можно поставить **load_example=false**:

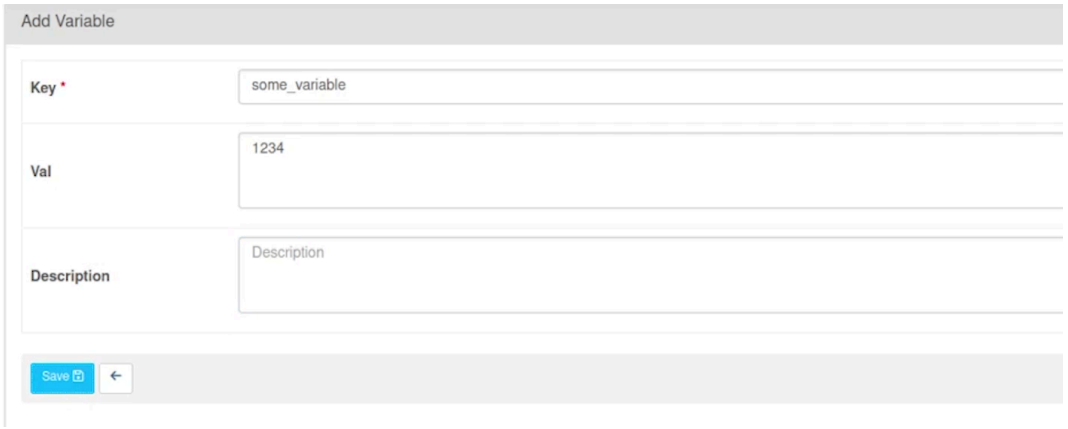


Мы посмотрели файл конфигурации:

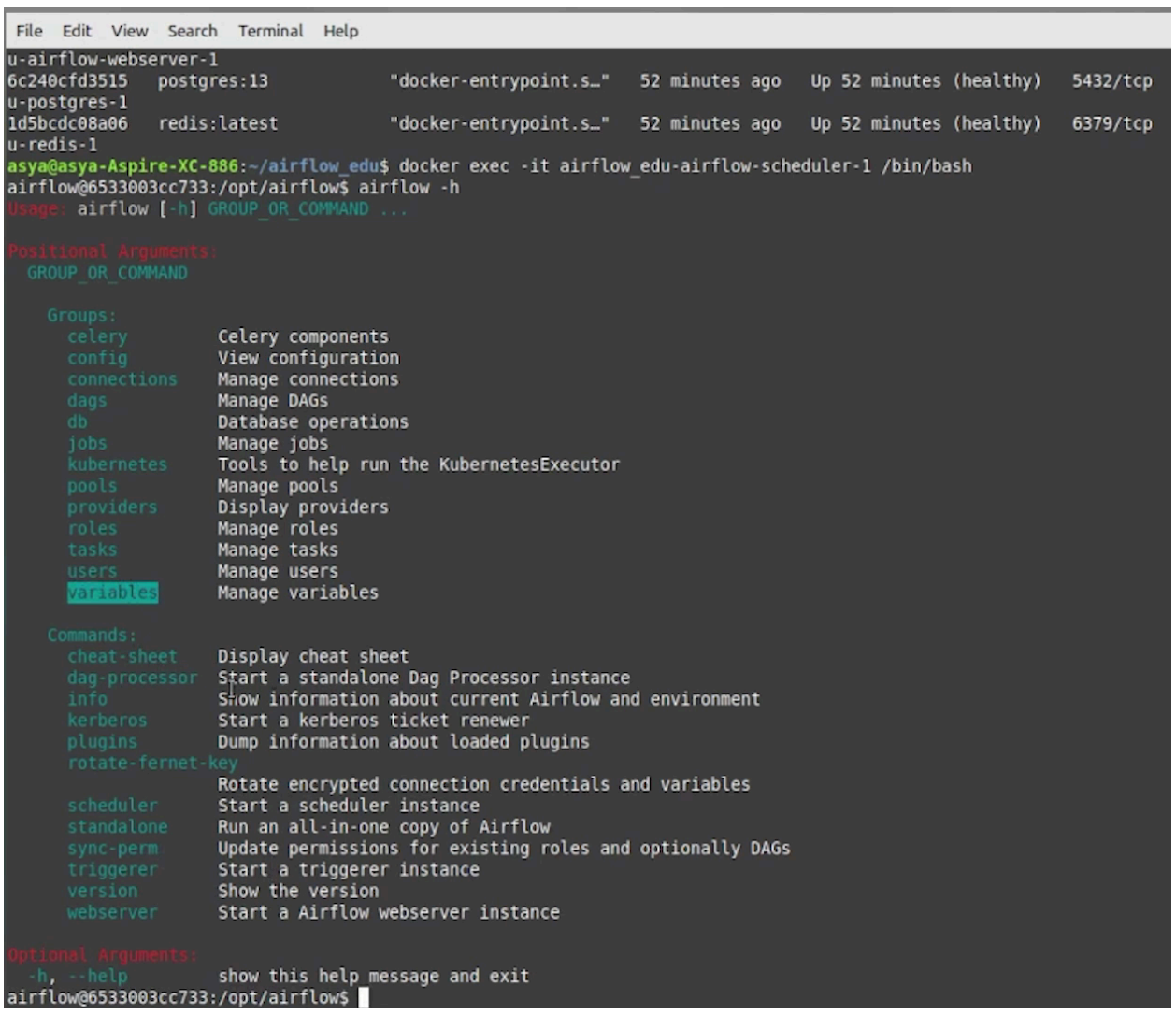


Посмотрим переменные, а именно, каким образом их можно задавать. Поскольку ранее мы их не задавали, для примера укажем ключ-значение. Желательно добавить описание по той же причине, что и connections, поскольку все variables хранятся в одном месте:

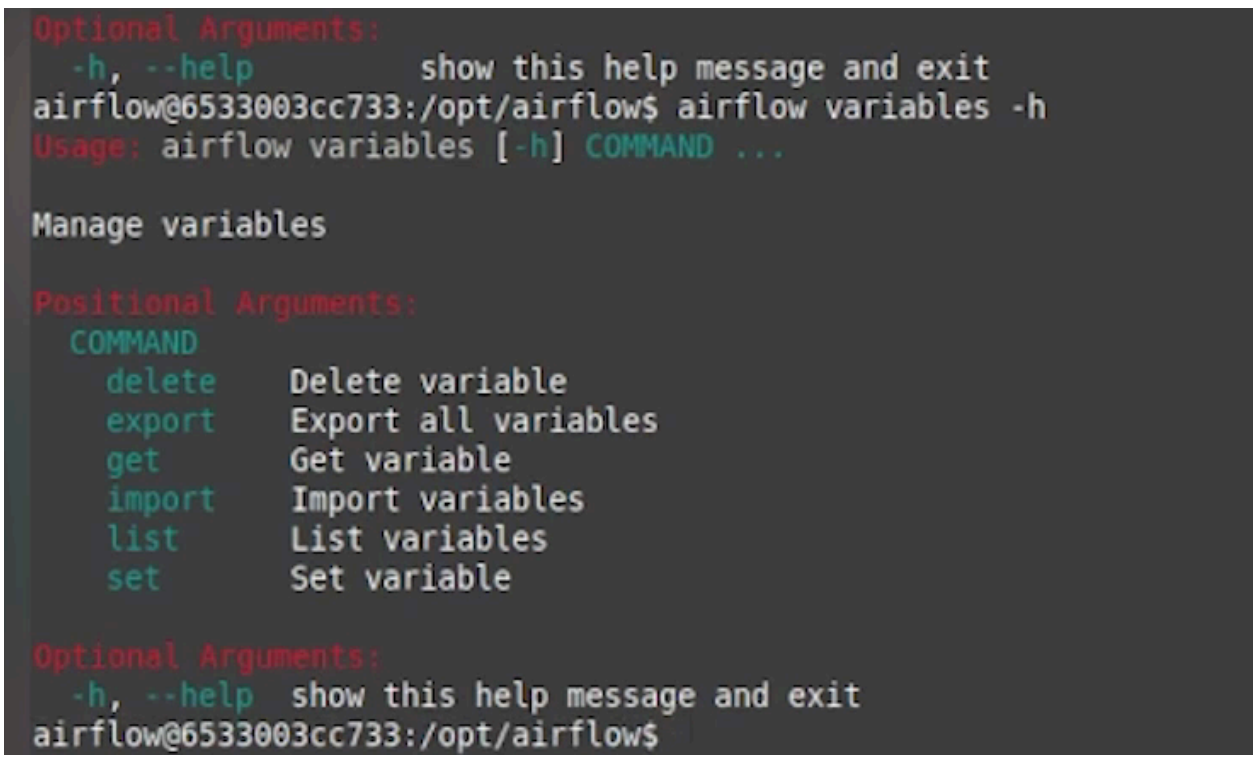
Дедлайн 04 августа, 23:59 Мск



Some variable должен сохраниться. Смотрим в Airflow (в контейнере «Scheduler». Здесь можно обращаться к variables:



Можно сделать лист всех переменных при помощи **variables -h**:



Можно установить некую новую переменную при помощи **new_war 4321**:

Дедлайн 04 августа, 23:59 Мск

```
Optional Arguments:
  -h, --help  show this help message and exit
airflow@6533003cc733:/opt/airflow$ airflow variables set new_var 4321
[2023-06-28T13:25:53.030+0000] {crypto.py:83} WARNING - empty cryptography key - values will not be stored encrypted.
Variable new_var created
airflow@6533003cc733:/opt/airflow$ airflow variables
Usage: airflow variables [-h] COMMAND ...

Manage variables

Positional Arguments:
  COMMAND
    delete    Delete variable
    export    Export all variables
    get       Get variable
    import    Import variables
    list      List variables
    set       Set variable

Optional Arguments:
  -h, --help  show this help message and exit

airflow variables command error: the following arguments are required: COMMAND, see help above.
airflow@6533003cc733:/opt/airflow$ airflow variables list
```

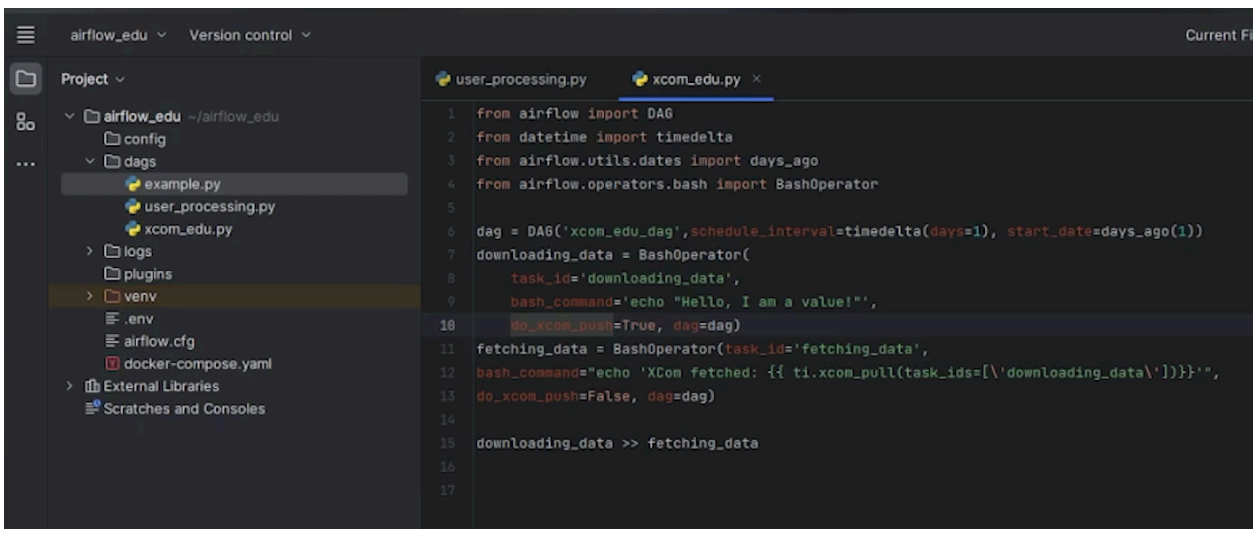
Мы видим два ключа: **some_variable и new_war**:

```
airflow variables command error: the following arguments are required: COMMAND, see help above.
airflow@6533003cc733:/opt/airflow$ airflow variables list
key
=============
some_variable
new_var
```

К ним можно обращаться из кода или получить здесь значение через get:

```
airflow@6533003cc733:/opt/airflow$ airflow variables get new_var
4321
airflow@6533003cc733:/opt/airflow$
```

Попробуем обратиться через код. Когда мы делаем variable.get, то получаем значения этой переменной. Таким образом выглядит получение переменной:



Variable.set – это установка.

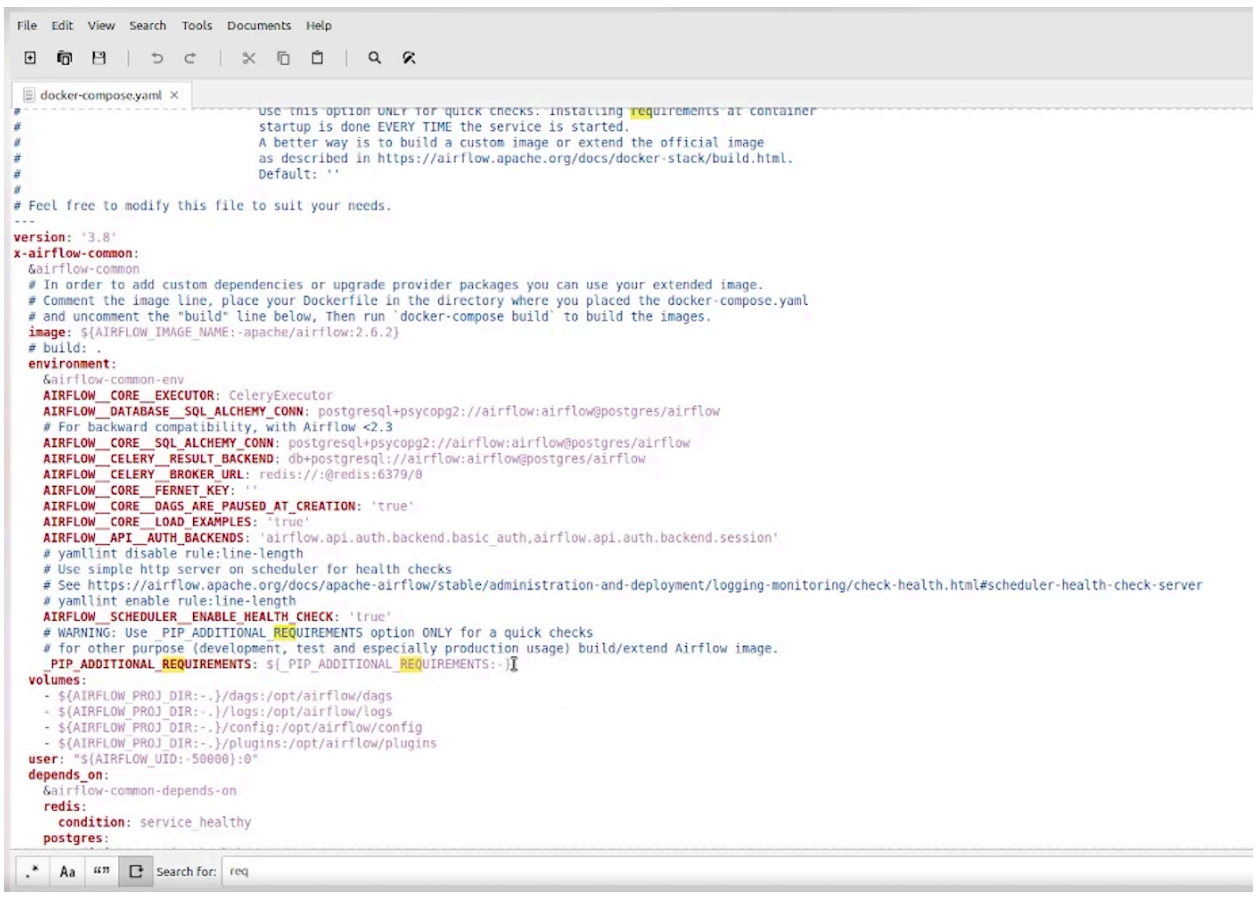
Через set можно установить значение, а через get поставить получателя.

Видим ту самую переменную «min_file-process_interval», указывающую интервал, с которого считывается значение:

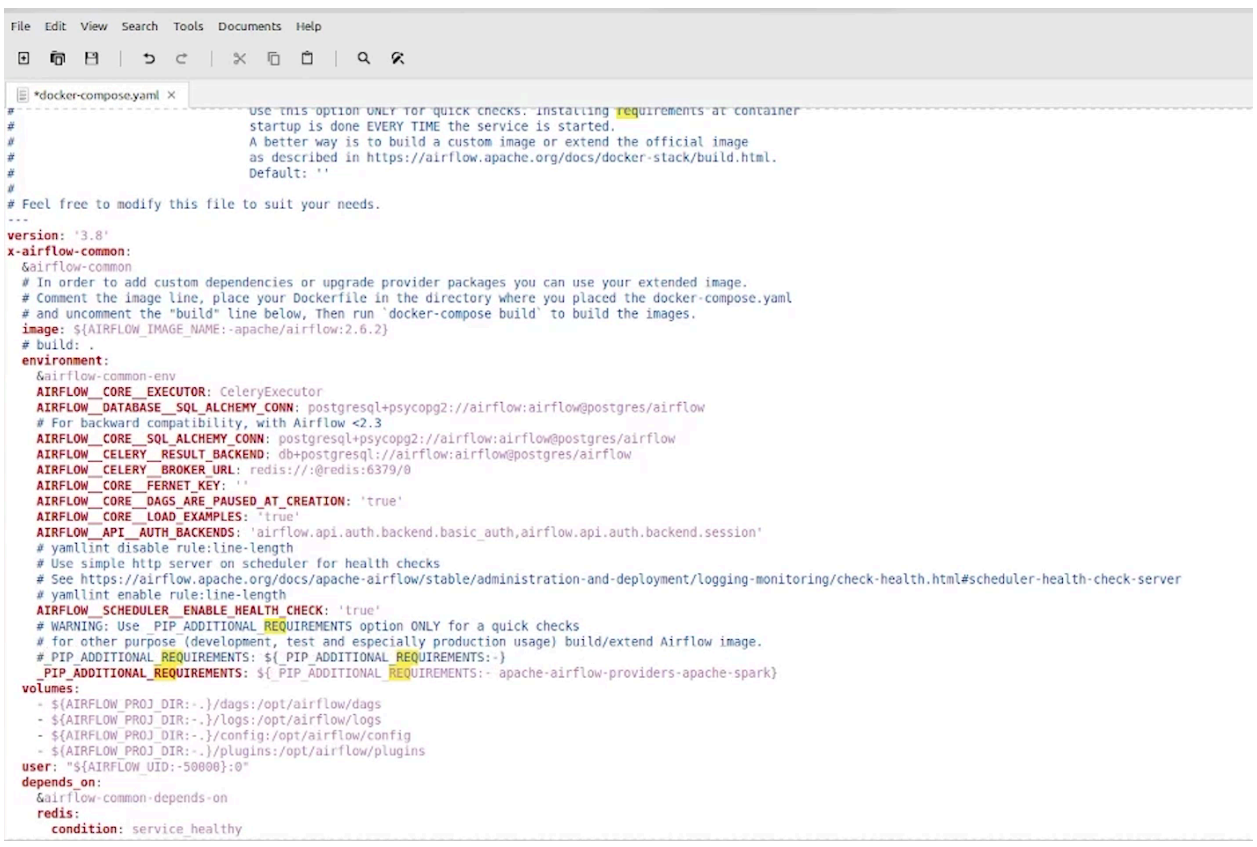
```
File  Edit  View  Search  Terminal  Help
  COMMAND
    delete    Delete variable
    export    Export all variables
    get       Get variable
    import    Import variables
    list      List variables
    set       Set variable

Optional Arguments:
  -h, --help  show this help message and exit
airflow@6533003cc733:/opt/airflow$ airflow variables set new_var 4321
[2023-06-28T13:25:53.030+0000] {crypto.py:83} WARNING - empty cryptography key - values will not be stored encrypted.
Variable new_var created
airflow@6533003cc733:/opt/airflow$ airflow variables
Usage: airflow variables [-h] COMMAND ...

Manage variables

Positional Arguments:
  COMMAND
    delete    Delete variable
    export    Export all variables
    get       Get variable
    import    Import variables
    list      List variables
    set       Set variable

Optional Arguments:
  -h, --help  show this help message and exit

airflow variables command error: the following arguments are required: COMMAND, see help above.
airflow@6533003cc733:/opt/airflow$ airflow variables list
key
============
some_variable
new_var

airflow@6533003cc733:/opt/airflow$ airflow variables get new_var
4321
airflow@6533003cc733:/opt/airflow$ ls
airflow.cfg  config  dags  logs  plugins  webserver_config.py
airflow@6533003cc733:/opt/airflow$ grep process_interval airflow.cfg
# ``min_file_process_interval`` number of seconds. Updates to DAGs are reflected after
min_file_process_interval = 30
airflow@6533003cc733:/opt/airflow$
```

Дедлайн 04 августа, 23:59 Мск

Рассмотрим файл «docker compose». Здесь мы можем указывать переменные. Например, в PIP_ADDTIONAL_REQUIREMENTS можно добавить дополнительные библиотеки, которые тоже нужно установить:



Делается это следующим образом:



Если мы перезапустим docker compose, то библиотека появится.

На этом уроке мы:

- Научились вносить изменения в файл конфигурации;
- Посмотрели из чего состоит файл конфигурации;
- Каким образом можно менять настройки;
- Как ставить новые библиотеки;
- Как устанавливать переменные.

Как вам урок?

Изучил, далее