СЛОЖНЫЕ ПАЙПЛАЙНЫ

KARPOV.COURSES

БЛОК ETL

1. Введение в ETL
2. Знакомство с Airflow

Вы находитесь здесь

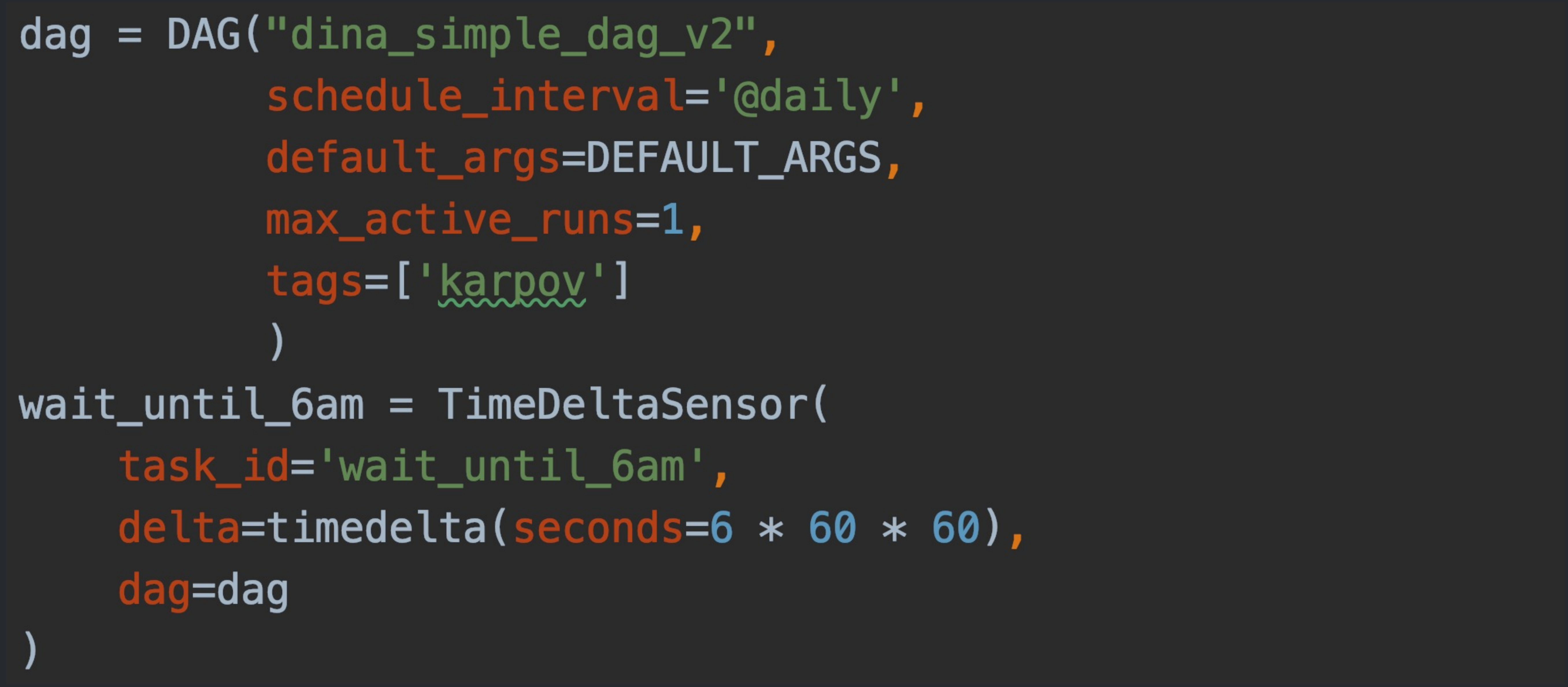
3. Сложные пайплайны, часть 1
4. Сложные пайплайны, часть 2
5. Разработка своих плагинов
6. Установка и настройка Airflow

ЛЕКЦИЯ «СЛОЖНЫЕ ПАЙПЛАЙНЫ»

1. Создание DAG’ов
2. Trigger Rule
3. Хуки, операторы и сенсоры
4. Ветвление
5. Шаблоны Jinja
6. Передача аргументов

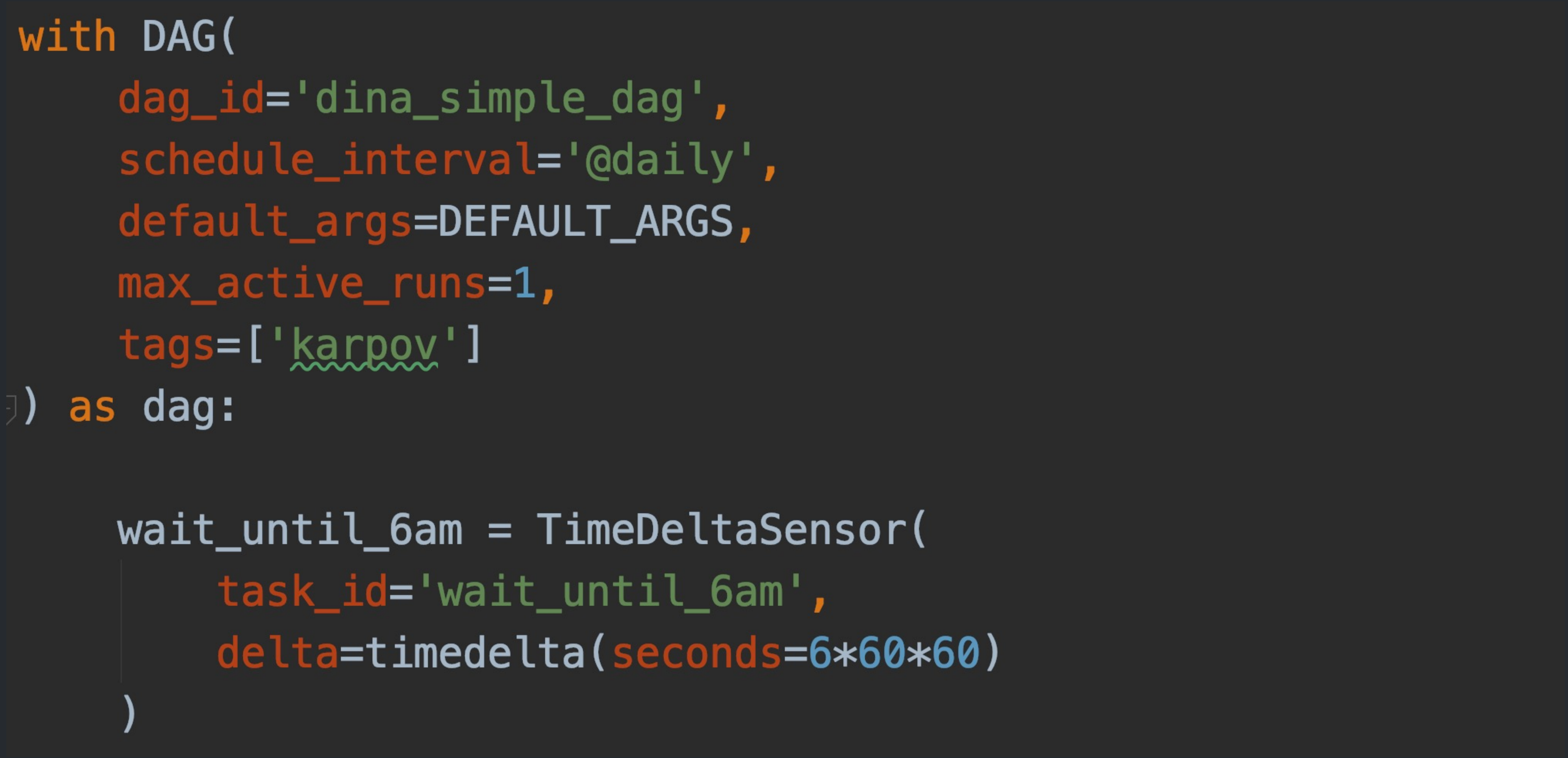
СОЗДАНИЕ DAG’A

СОЗДАНИЕ DAG'A



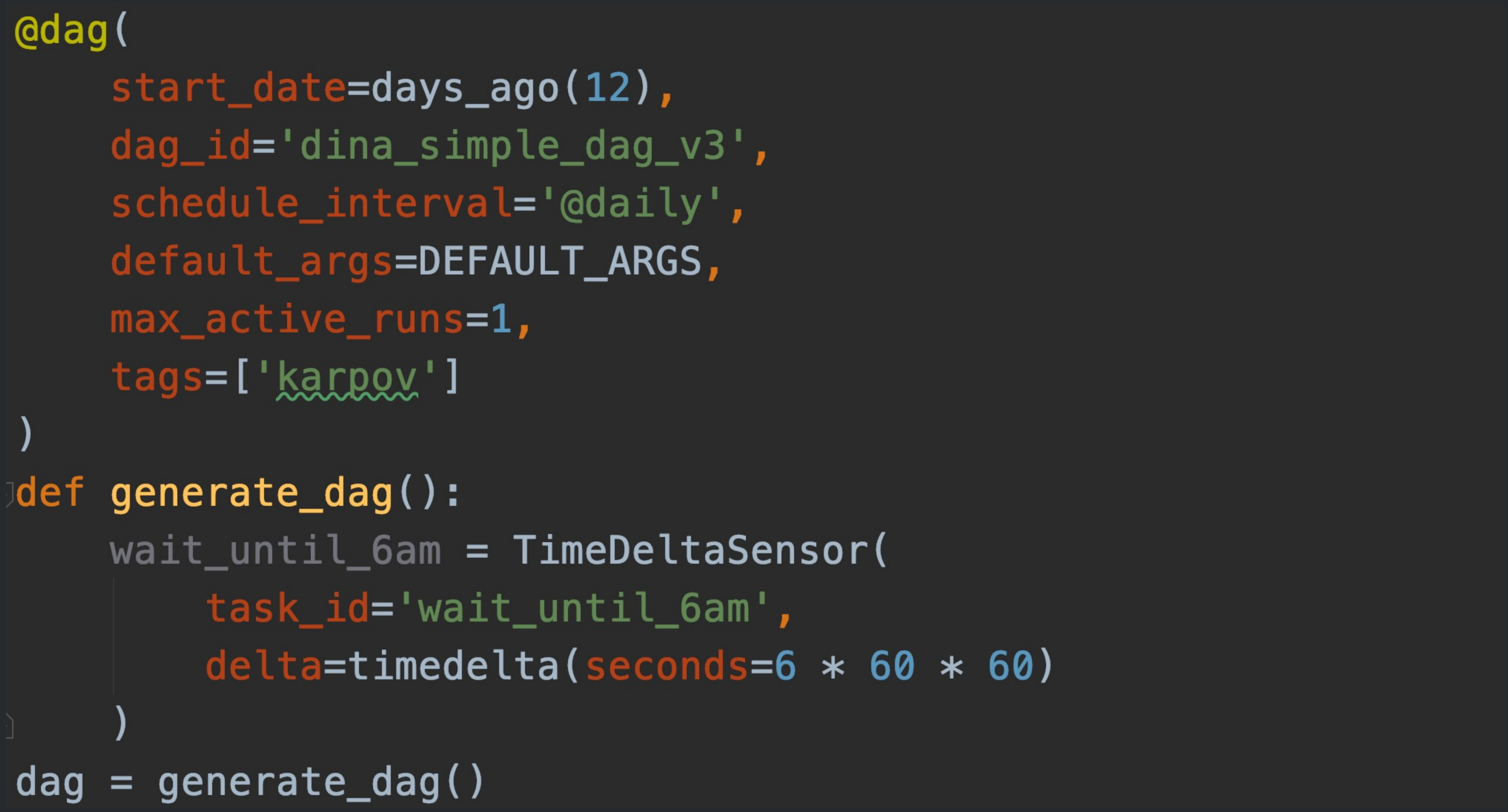
```
dag = DAG("dina_simple_dag_v2",
          schedule_interval='@daily',
          default_args=DEFAULT_ARGS,
          max_active_runs=1,
          tags=['karpov']
          )
wait_until_6am = TimeDeltaSensor(
    task_id='wait_until_6am',
    delta=timedelta(seconds=6 * 60 * 60),
    dag=dag
)
```

СОЗДАНИЕ DAG’A. ВАРИАНТ 2

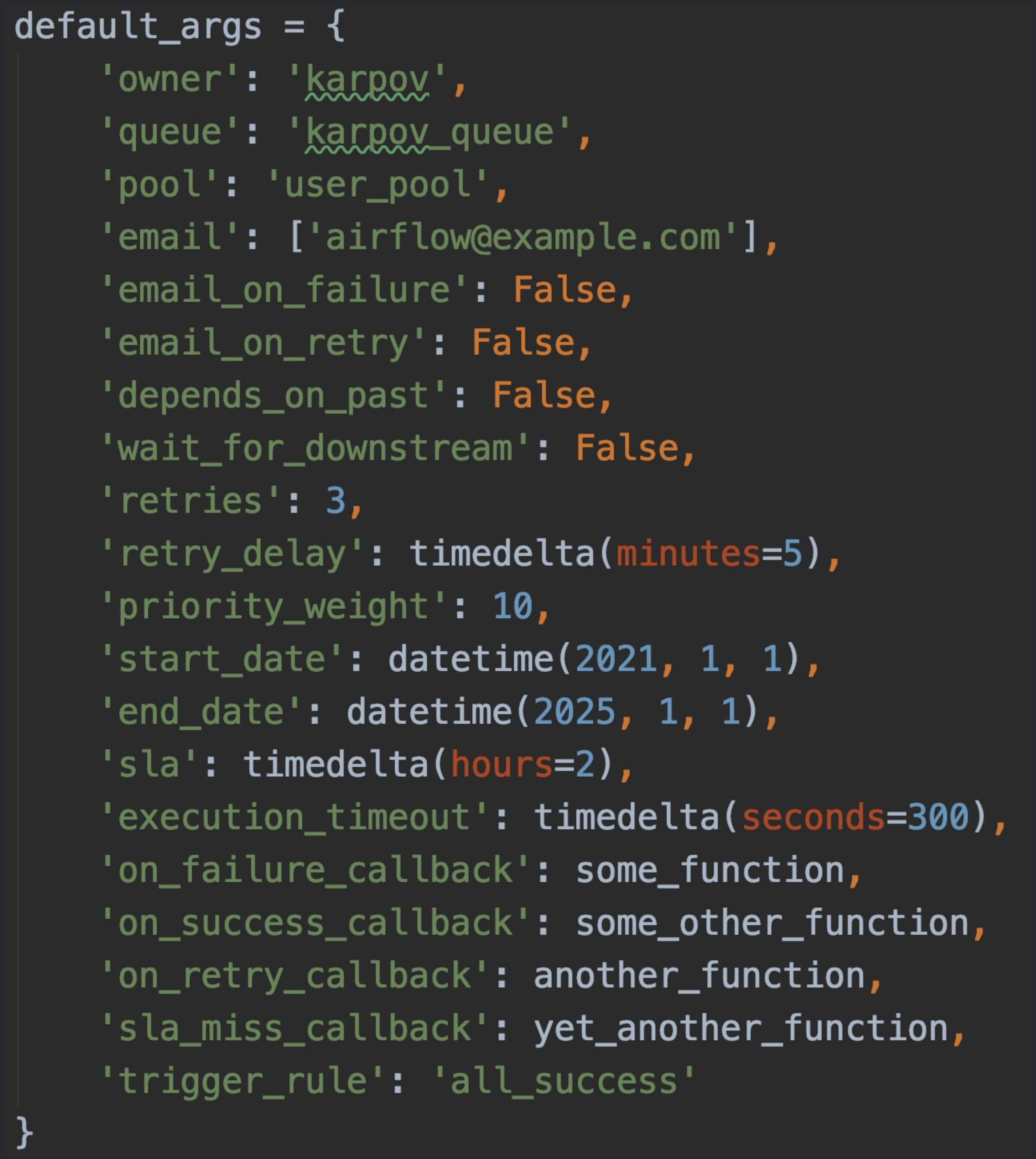


```
with DAG(
    dag_id='dina_simple_dag',
    schedule_interval='@daily',
    default_args=DEFAULT_ARGS,
    max_active_runs=1,
    tags=['karpov']
) as dag:

    wait_until_6am = TimeDeltaSensor(
        task_id='wait_until_6am',
        delta=timedelta(seconds=6*60*60)
    )
```

СОЗДАНИЕ DAG’A. ВАРИАНТ 3



```
@dag(
    start_date=days_ago(12),
    dag_id='dina_simple_dag_v3',
    schedule_interval='@daily',
    default_args=DEFAULT_ARGS,
    max_active_runs=1,
    tags=['karpov']
)
def generate_dag():
    wait_until_6am = TimeDeltaSensor(
        task_id='wait_until_6am',
        delta=timedelta(seconds=6 * 60 * 60)
    )
dag = generate_dag()
```

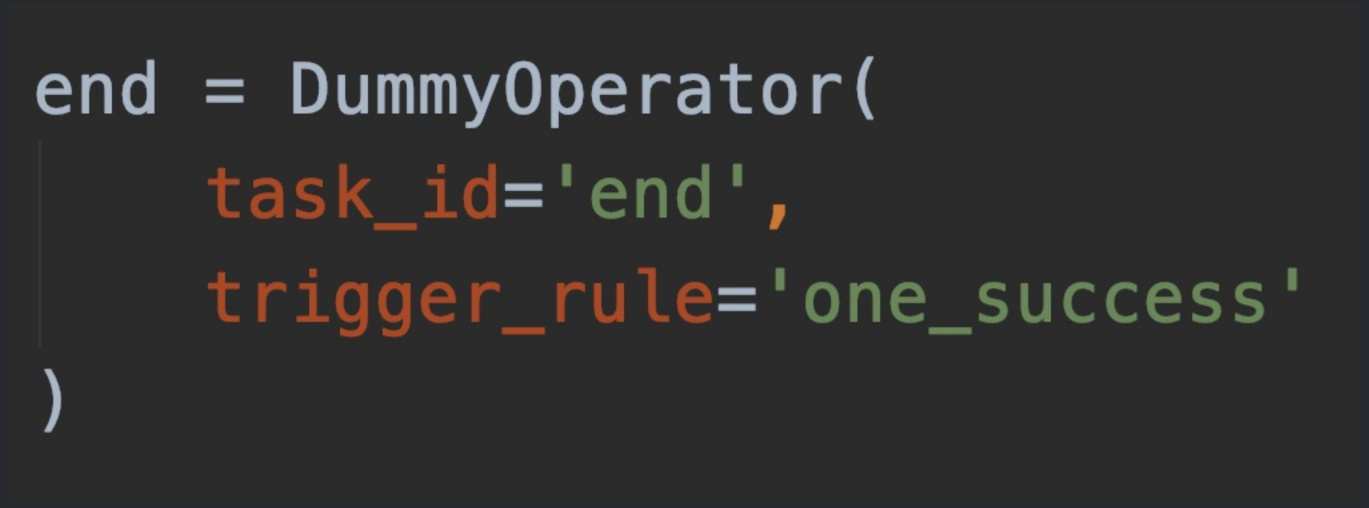


```
default_args = {
    'owner': 'karpov',
    'queue': 'karpov_queue',
    'pool': 'user_pool',
    'email': ['airflow@example.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'depends_on_past': False,
    'wait_for_downstream': False,
    'retries': 3,
    'retry_delay': timedelta(minutes=5),
    'priority_weight': 10,
    'start_date': datetime(2021, 1, 1),
    'end_date': datetime(2025, 1, 1),
    'sla': timedelta(hours=2),
    'execution_timeout': timedelta(seconds=300),
    'on_failure_callback': some_function,
    'on_success_callback': some_other_function,
    'on_retry_callback': another_function,
    'sla_miss_callback': yet_another_function,
    'trigger_rule': 'all_success'
}
```

TRIGGER RULE

TRIGGER RULE

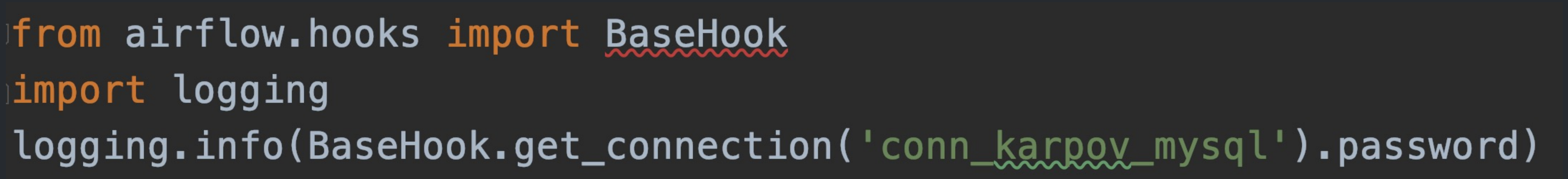
- all_success
- all_failed
- all_done
- one_failed
- one_success
- none_failed
- none_failed_or_skipped
- none_skipped
- dummy



```
end = DummyOperator(
    task_id='end',
    trigger_rule='one_success'
)
```

ХУКИ, ОПЕРАТОРЫ, СЕНСОРЫ

CONNECTIONS



```
from airflow.hooks import BaseHook
import logging
logging.info(BaseHook.get_connection('conn_karpov_mysql').password)
```

HOOKS

— S3Hook

— DockerHook

— HDFSHook

— HttpHook

— MsSqlHook

— MySqlHook

— OracleHook

— PigCliHook

— PostgresHook

— SqliteHook

ОПЕРАТОРЫ

— BashOperator
— PythonOperator
— EmailOperator
— PostgresOperator
— MySqlOperator
— MsSqlOperator
— HiveOperator
— SimpleHttpOperator
— SlackAPIOperator
— PrestoToMySqlOperator
— TriggerDagRunOperator

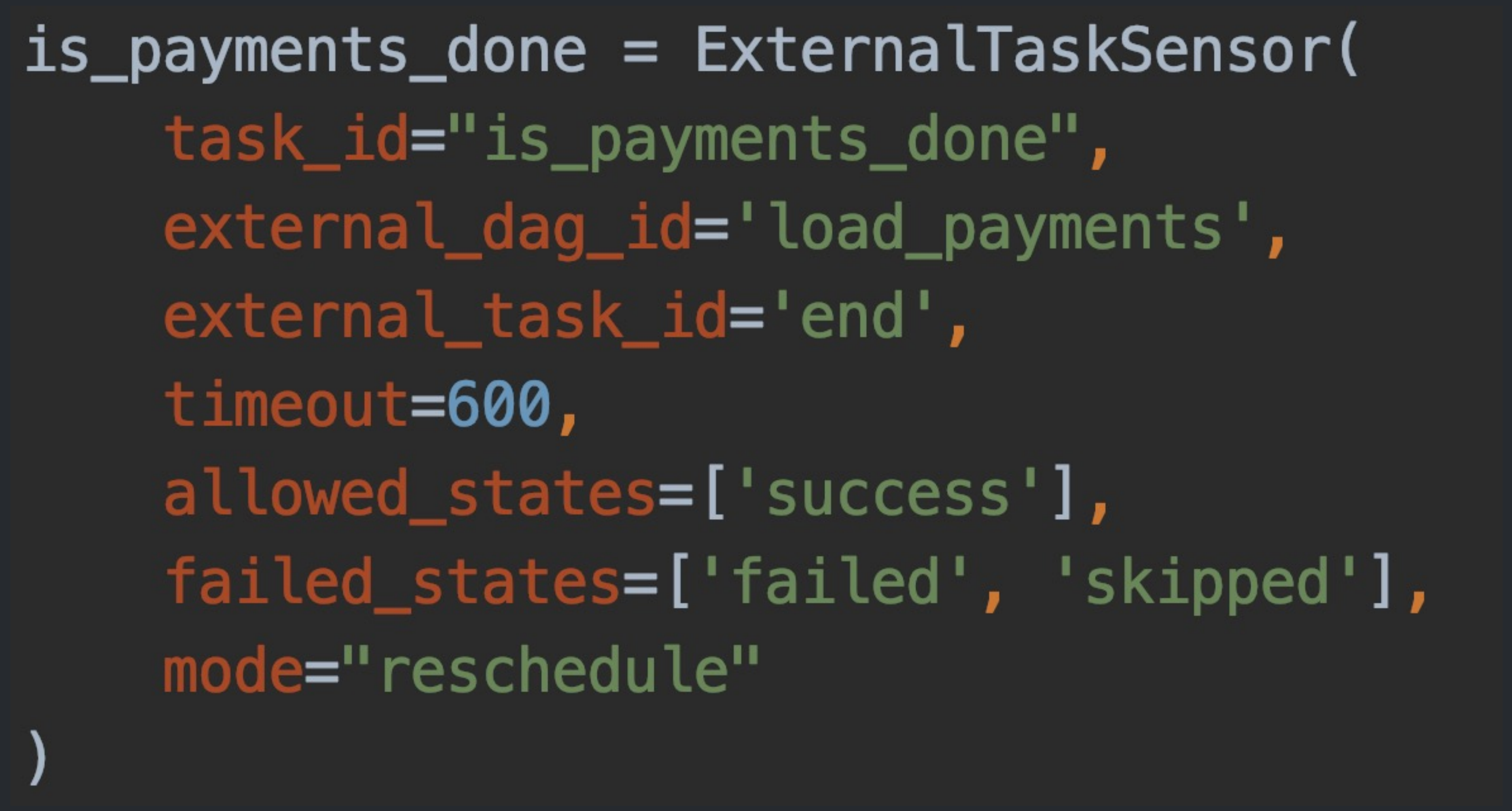
СЕНСОРЫ

— timeout
— soft_fail
— poke_interval
— mode — poke | reschedule

СЕНСОРЫ

— ExternalTaskSensor
— SqlSensor
— TimeDeltaSensor
— HdfsSensor
— PythonSensor
— DayOfWeekSensor

ExternalTaskSensor



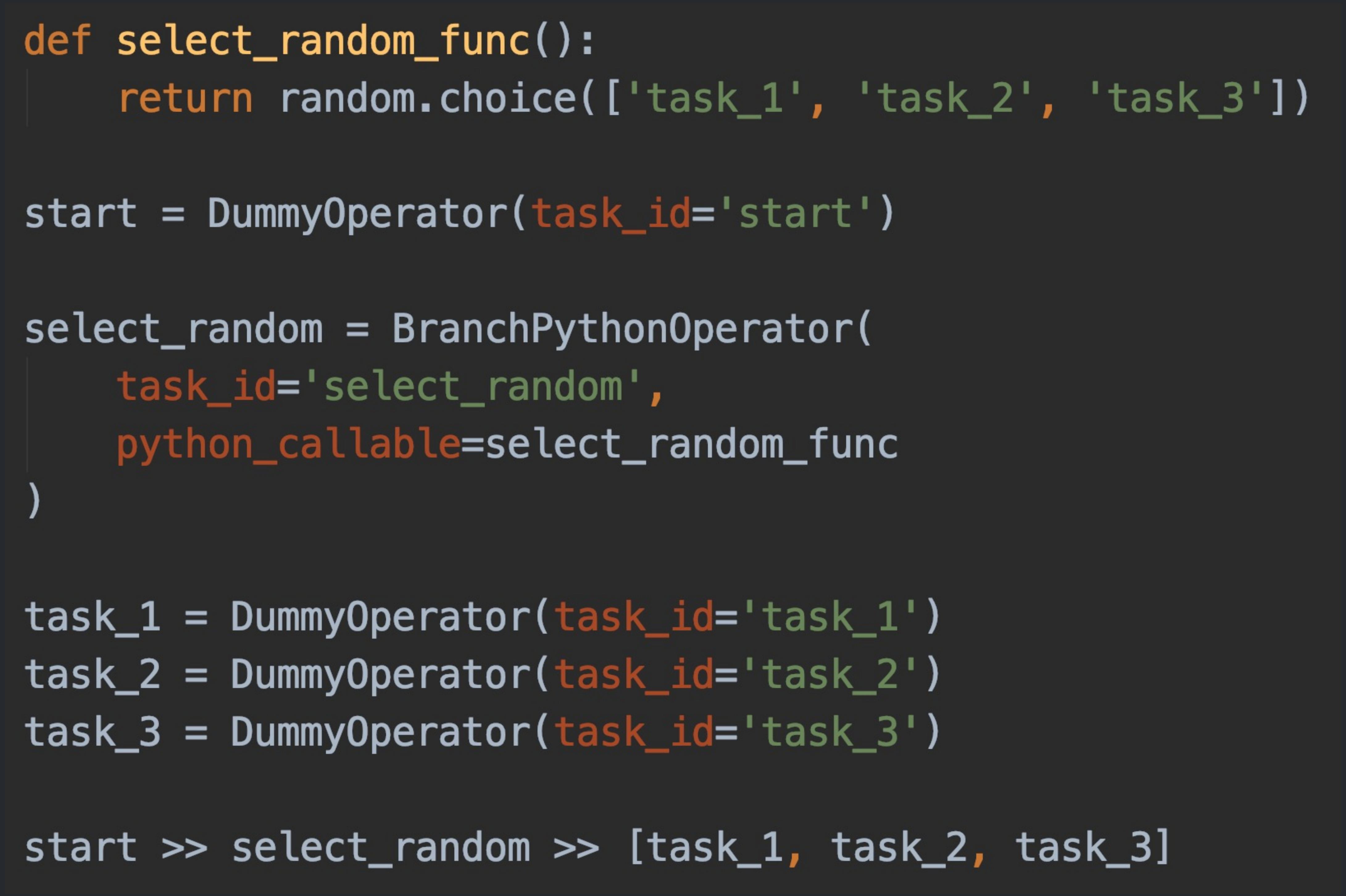
```
is_payments_done = ExternalTaskSensor(
    task_id="is_payments_done",
    external_dag_id='load_payments',
    external_task_id='end',
    timeout=600,
    allowed_states=['success'],
    failed_states=['failed', 'skipped'],
    mode="reschedule"
)
```

ВЕТВЛЕНИЕ

ВЕТВЛЕНИЕ

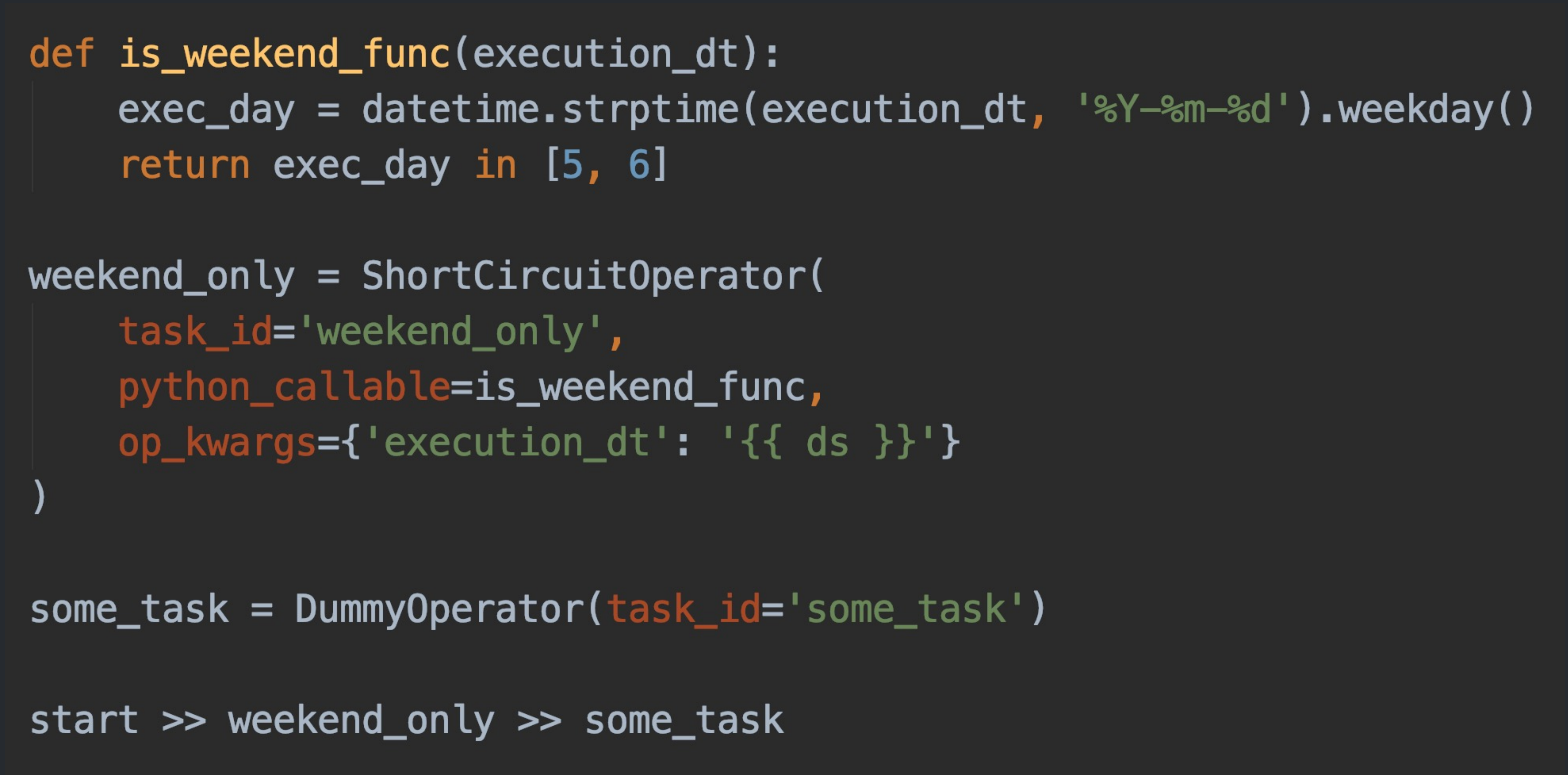
— BranchPythonOperator
— ShortCircuitOperator
— BrachDateTimeOperator

BranchPythonOperator

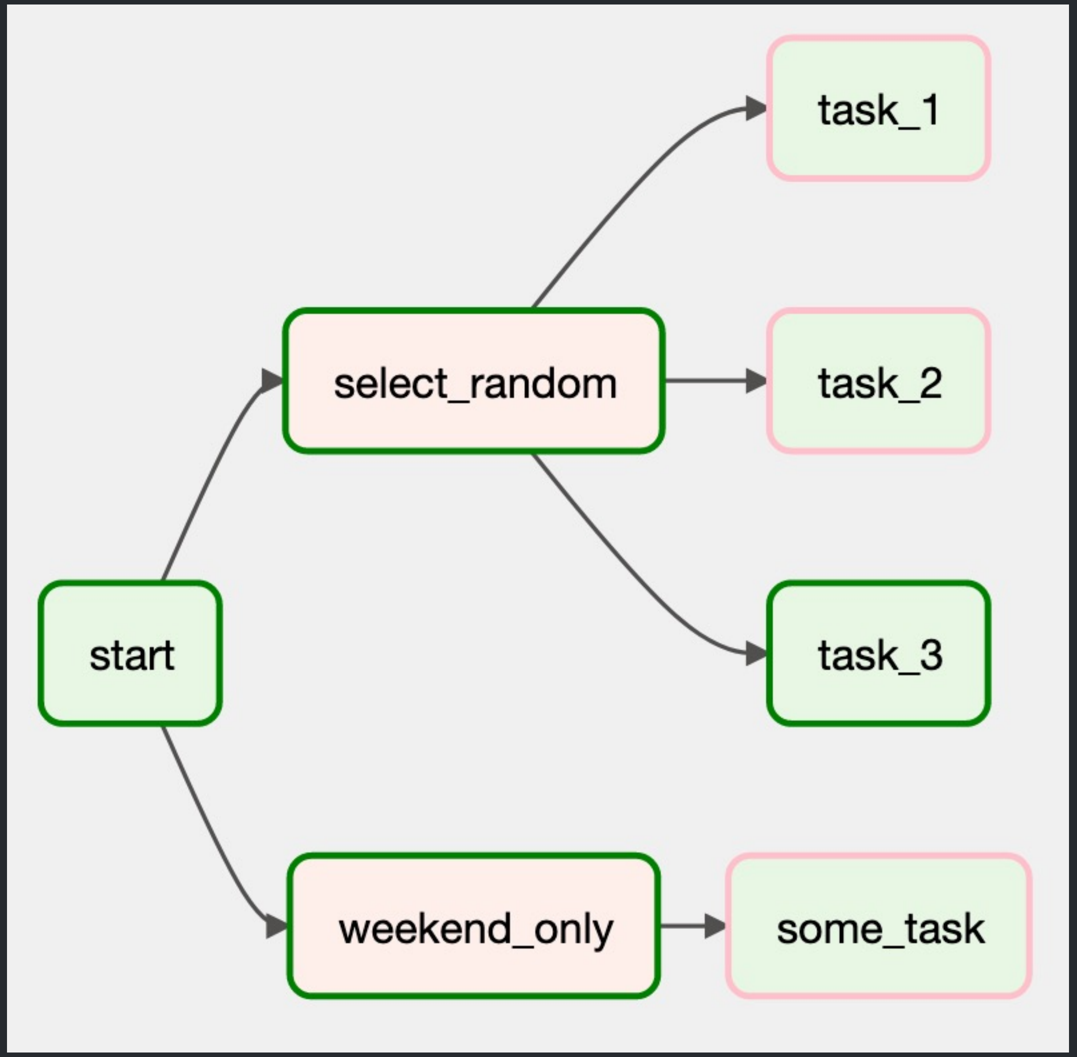


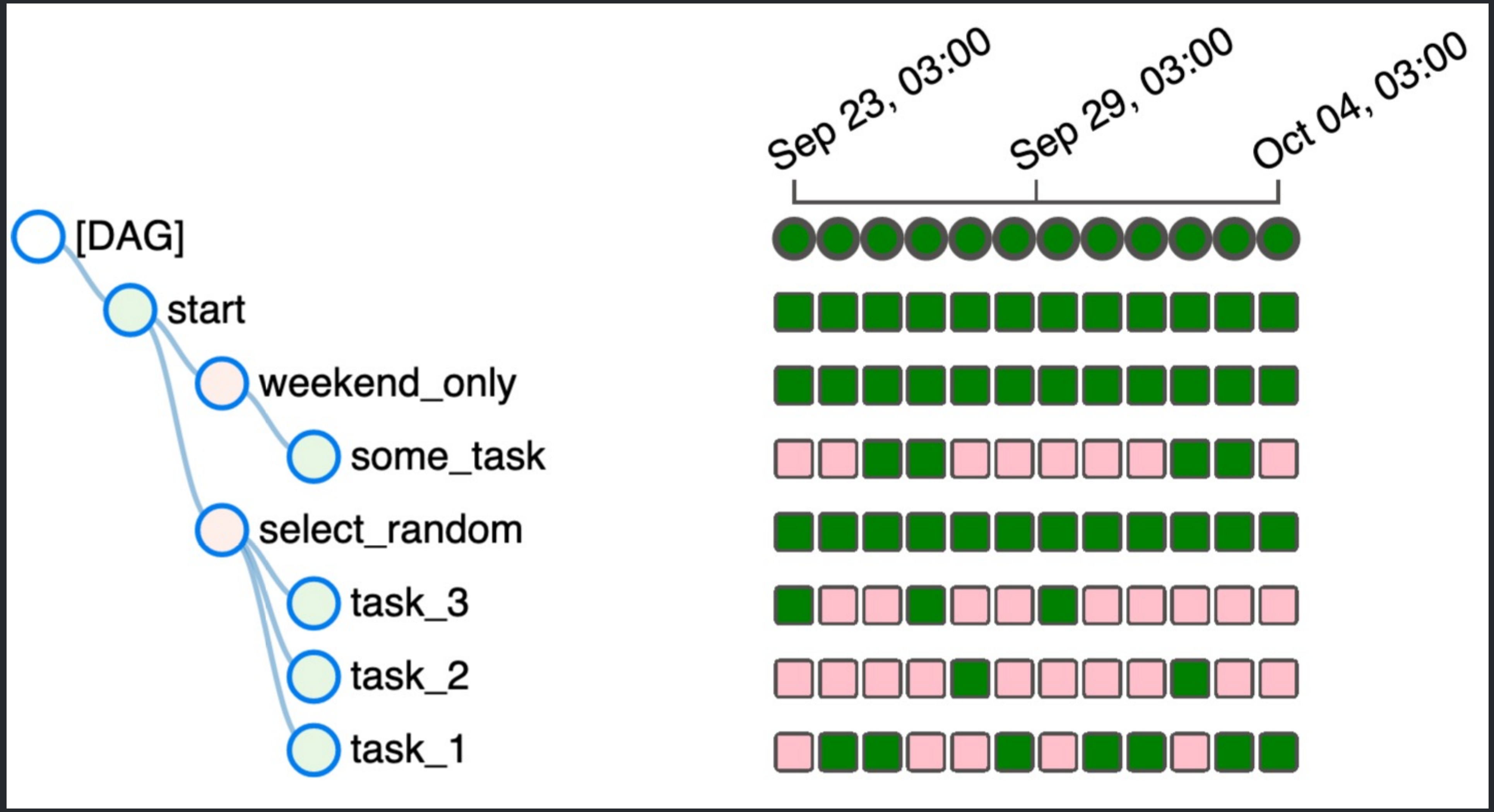
```python
def select_random_func():
    return random.choice(['task_1', 'task_2', 'task_3'])

start = DummyOperator(task_id='start')

select_random = BranchPythonOperator(
    task_id='select_random',
    python_callable=select_random_func
)

task_1 = DummyOperator(task_id='task_1')
task_2 = DummyOperator(task_id='task_2')
task_3 = DummyOperator(task_id='task_3')

start >> select_random >> [task_1, task_2, task_3]
```

ShortCircuitOperator



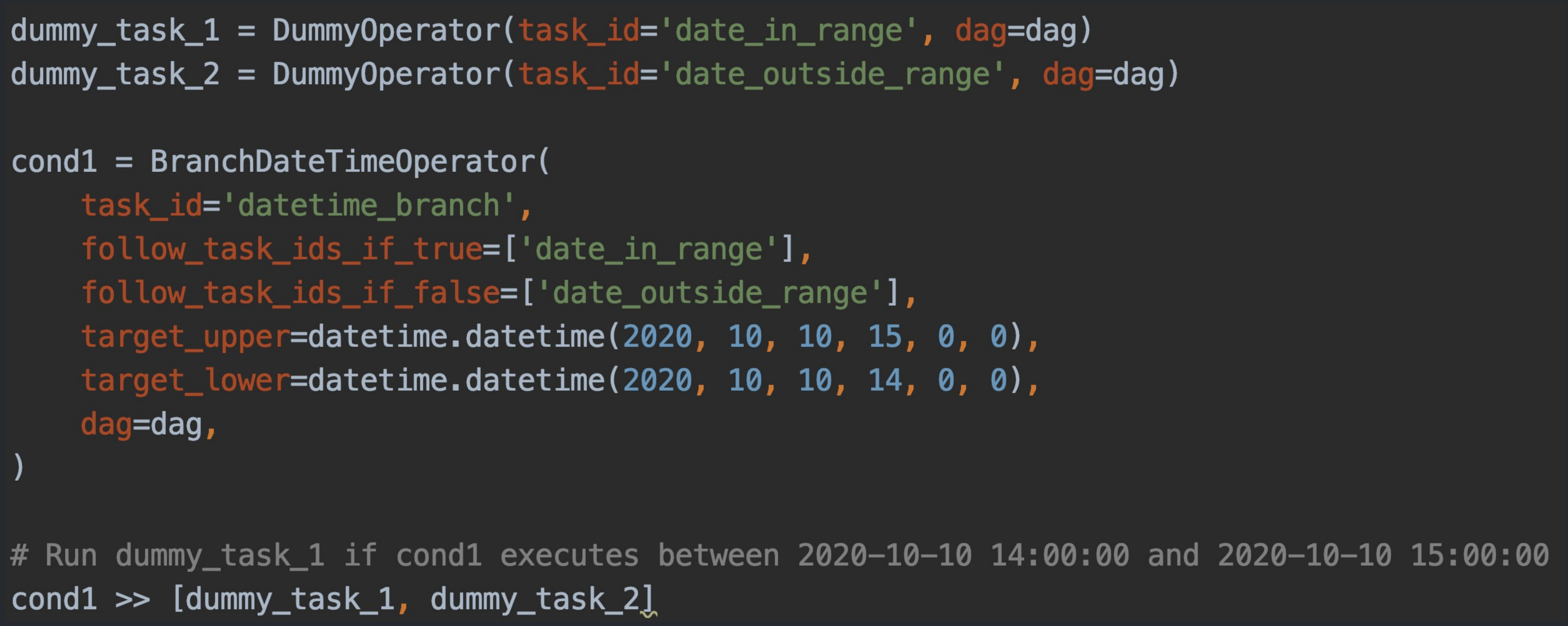
```python
def is_weekend_func(execution_dt):
    exec_day = datetime.strptime(execution_dt, '%Y-%m-%d').weekday()
    return exec_day in [5, 6]

weekend_only = ShortCircuitOperator(
    task_id='weekend_only',
    python_callable=is_weekend_func,
    op_kwargs={'execution_dt': '{{ ds }}'}
)

some_task = DummyOperator(task_id='some_task')

start >> weekend_only >> some_task
```





[DAG]	●	●	●	●	●	●	●	●	●	●	●	●
start	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
weekend_only	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
some_task	🟥	🟥	🟩	🟩	🟥	🟥	🟥	🟥	🟥	🟩	🟩	🟥
select_random	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
task_3	🟩	🟥	🟥	🟩	🟥	🟥	🟩	🟥	🟥	🟥	🟥	🟥
task_2	🟥	🟥	🟥	🟥	🟩	🟥	🟥	🟥	🟥	🟩	🟥	🟥
task_1	🟥	🟩	🟩	🟥	🟥	🟩	🟥	🟩	🟩	🟥	🟩	🟩

BranchDateTimeOperator



```python
dummy_task_1 = DummyOperator(task_id='date_in_range', dag=dag)
dummy_task_2 = DummyOperator(task_id='date_outside_range', dag=dag)

cond1 = BranchDateTimeOperator(
    task_id='datetime_branch',
    follow_task_ids_if_true=['date_in_range'],
    follow_task_ids_if_false=['date_outside_range'],
    target_upper=datetime.datetime(2020, 10, 10, 15, 0, 0),
    target_lower=datetime.datetime(2020, 10, 10, 14, 0, 0),
    dag=dag,
)

# Run dummy_task_1 if cond1 executes between 2020-10-10 14:00:00 and 2020-10-10 15:00:00
cond1 >> [dummy_task_1, dummy_task_2]
```

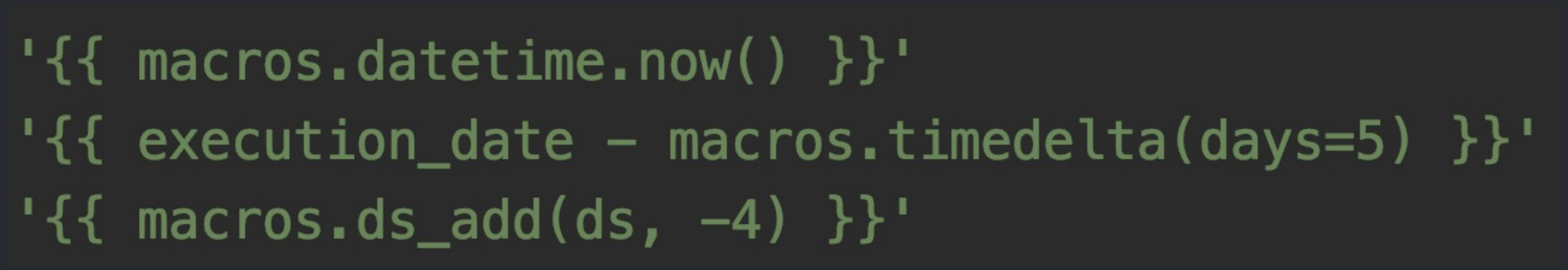
ШАБЛОНЫ JINJA

ШАБЛОНИЗАЦИЯ

Шаблон	Расшифровка
{{ execution_date }}	execution_date
{{ ds }}	execution_date (YYYY-MM-DD)
{{ ds_nodash }}	execution_date (YYYYMMDD)
{{ ts }}	execution_date (2021-01-01T00:00:00+00:00)
{{ yesterday_ds }}	Вчерашний день относительно execution_date
{{ tomorrow_ds }}	Завтрашний день относительно execution_date
{{ var.value.my_var }}	Значение ключа в глобальной переменной (словарь)
{{ var.json.my_var.path }}	Значение ключа в глобальной переменной (json)
{{ conf }}	airflow.cfg

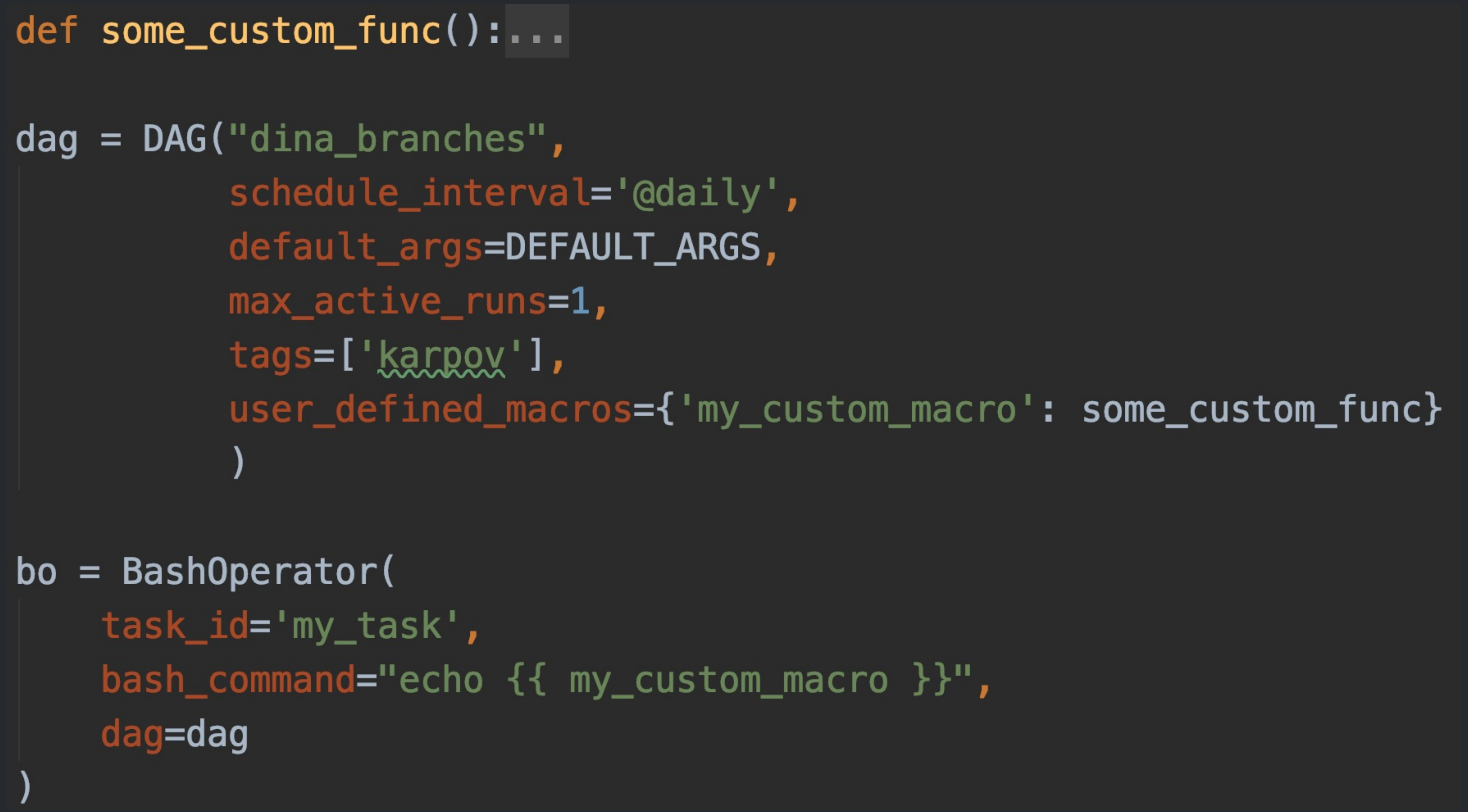
МАКРОСЫ

Переменная	Пакет в Питоне
macros.datetime	datetime.datetime
macros.timedelta	datetime.timedelta
macros.dateutil	dateutil
macros.time	datetime.time
macros.uuid	uuid
macros.random	random

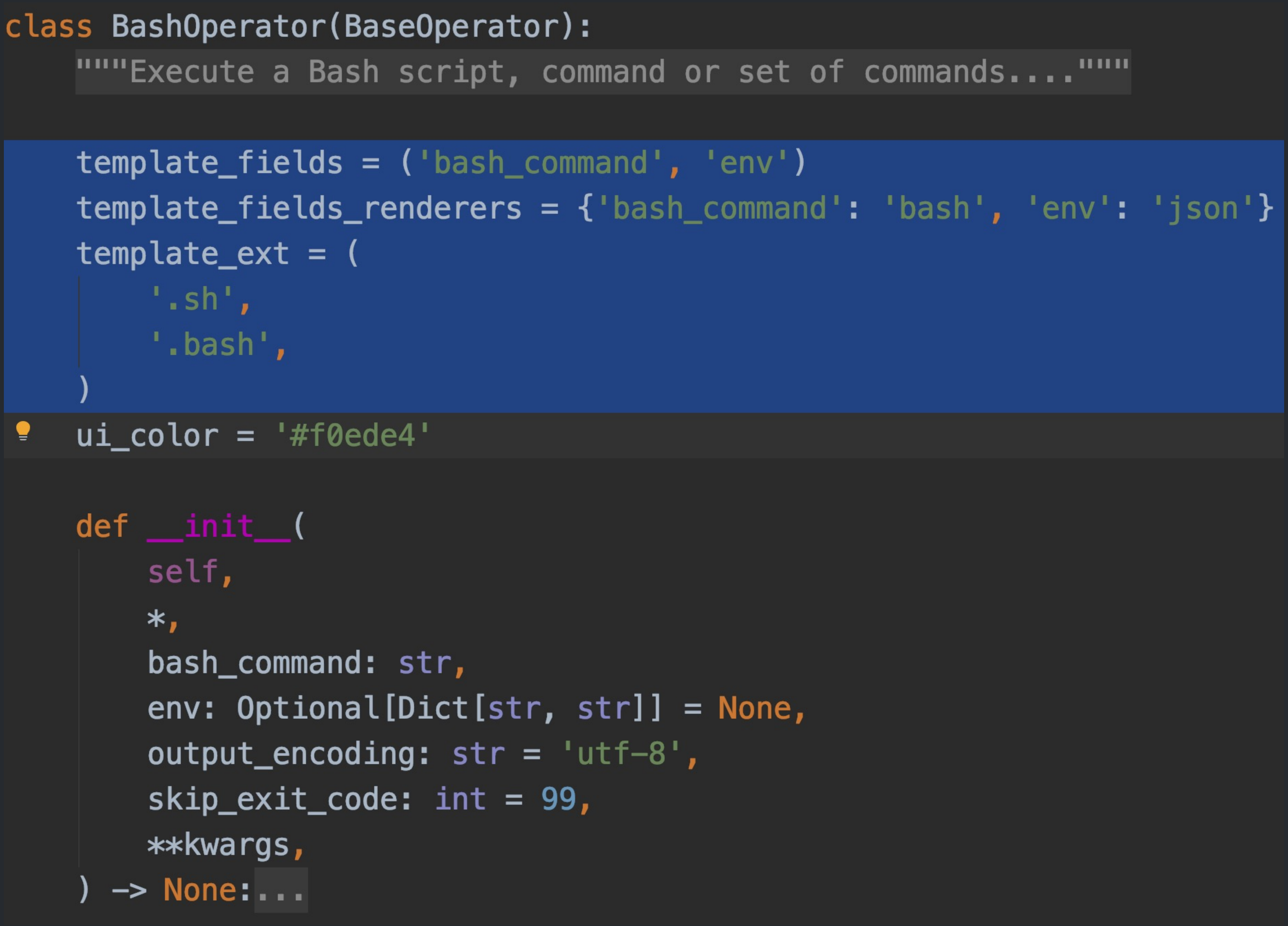


```
'{{ macros.datetime.now() }}'
'{{ execution_date - macros.timedelta(days=5) }}'
'{{ macros.ds_add(ds, -4) }}'
```

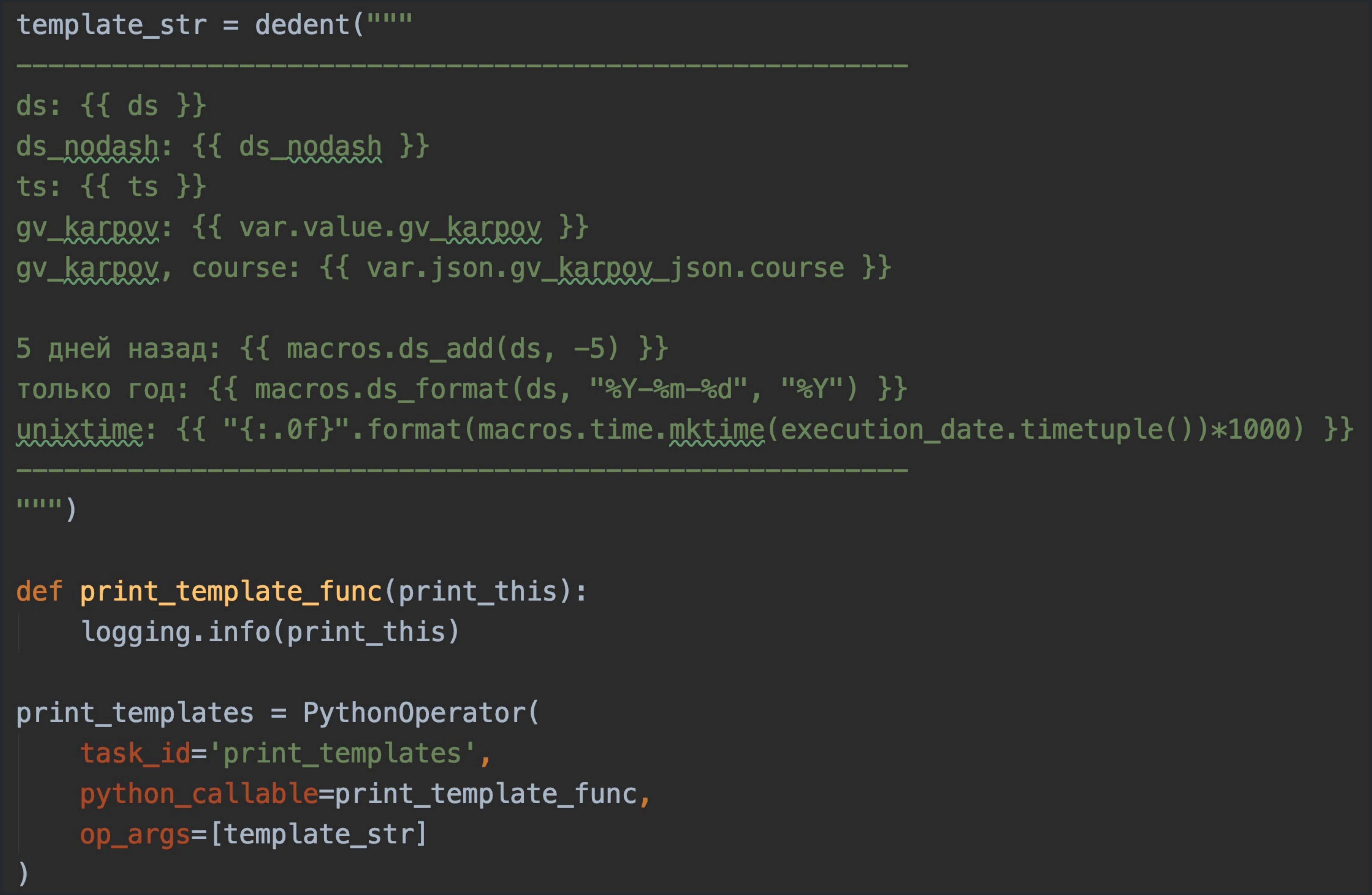
ПОЛЬЗОВАТЕЛЬСКИЕ МАКРОСЫ



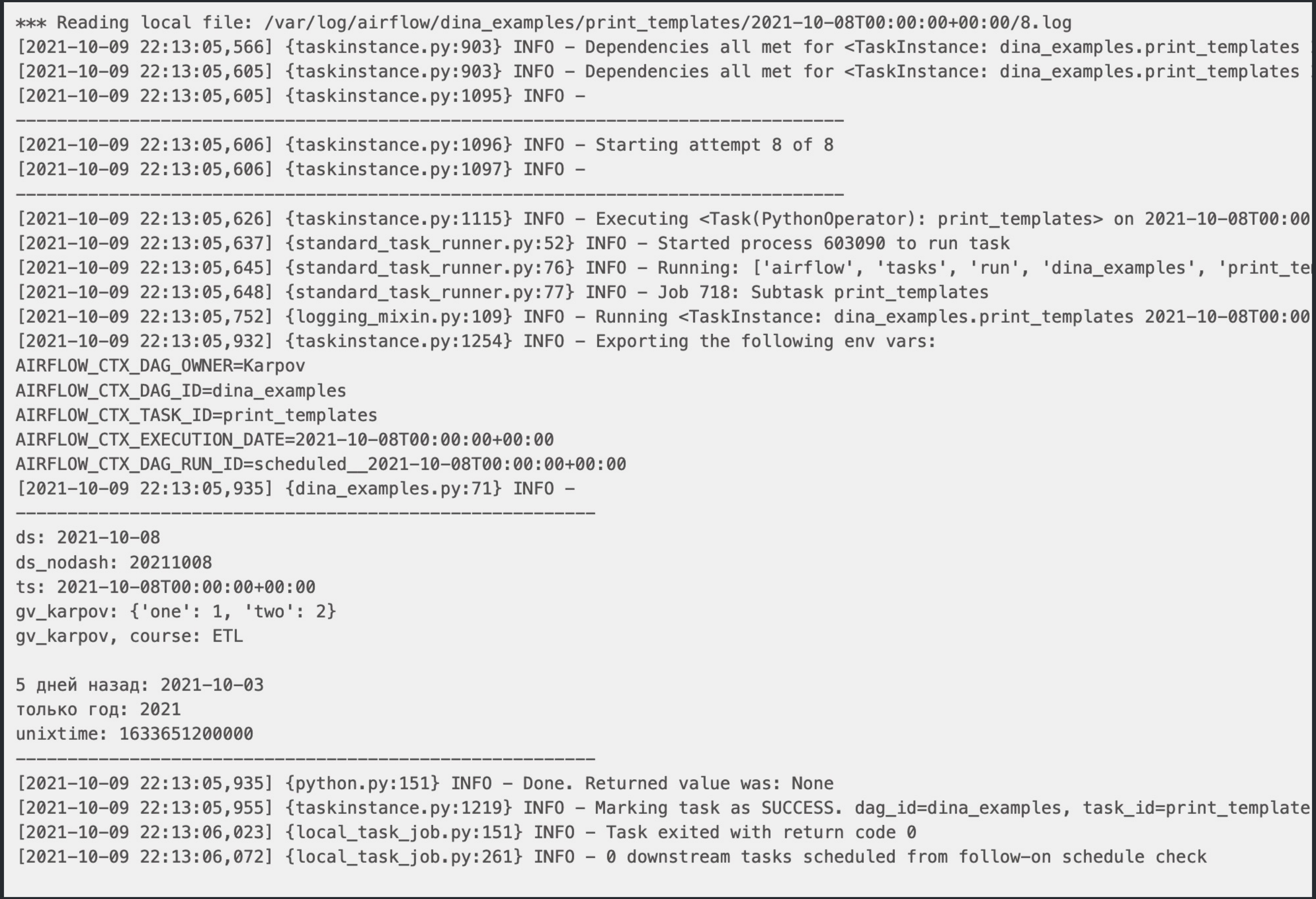
```python
def some_custom_func():...

dag = DAG("dina_branches",
          schedule_interval='@daily',
          default_args=DEFAULT_ARGS,
          max_active_runs=1,
          tags=['karpov'],
          user_defined_macros={'my_custom_macro': some_custom_func}
          )

bo = BashOperator(
    task_id='my_task',
    bash_command="echo {{ my_custom_macro }}",
    dag=dag
)
```



```python
class BashOperator(BaseOperator):
    """Execute a Bash script, command or set of commands...."""

    template_fields = ('bash_command', 'env')
    template_fields_renderers = {'bash_command': 'bash', 'env': 'json'}
    template_ext = (
        '.sh',
        '.bash',
    )
    ui_color = '#f0ede4'

    def __init__(
        self,
        *,
        bash_command: str,
        env: Optional[Dict[str, str]] = None,
        output_encoding: str = 'utf-8',
        skip_exit_code: int = 99,
        **kwargs,
    ) -> None:...
```



```
template_str = dedent("""
--------------------------------------------------------
ds: {{ ds }}
ds_nodash: {{ ds_nodash }}
ts: {{ ts }}
gv_karpov: {{ var.value.gv_karpov }}
gv_karpov, course: {{ var.json.gv_karpov_json.course }}

5 дней назад: {{ macros.ds_add(ds, -5) }}
только год: {{ macros.ds_format(ds, "%Y-%m-%d", "%Y") }}
unixtime: {{ "{:.0f}".format(macros.time.mktime(execution_date.timetuple())*1000) }}
--------------------------------------------------------
""")


def print_template_func(print_this):
    logging.info(print_this)


print_templates = PythonOperator(
    task_id='print_templates',
    python_callable=print_template_func,
    op_args=[template_str]
)
```

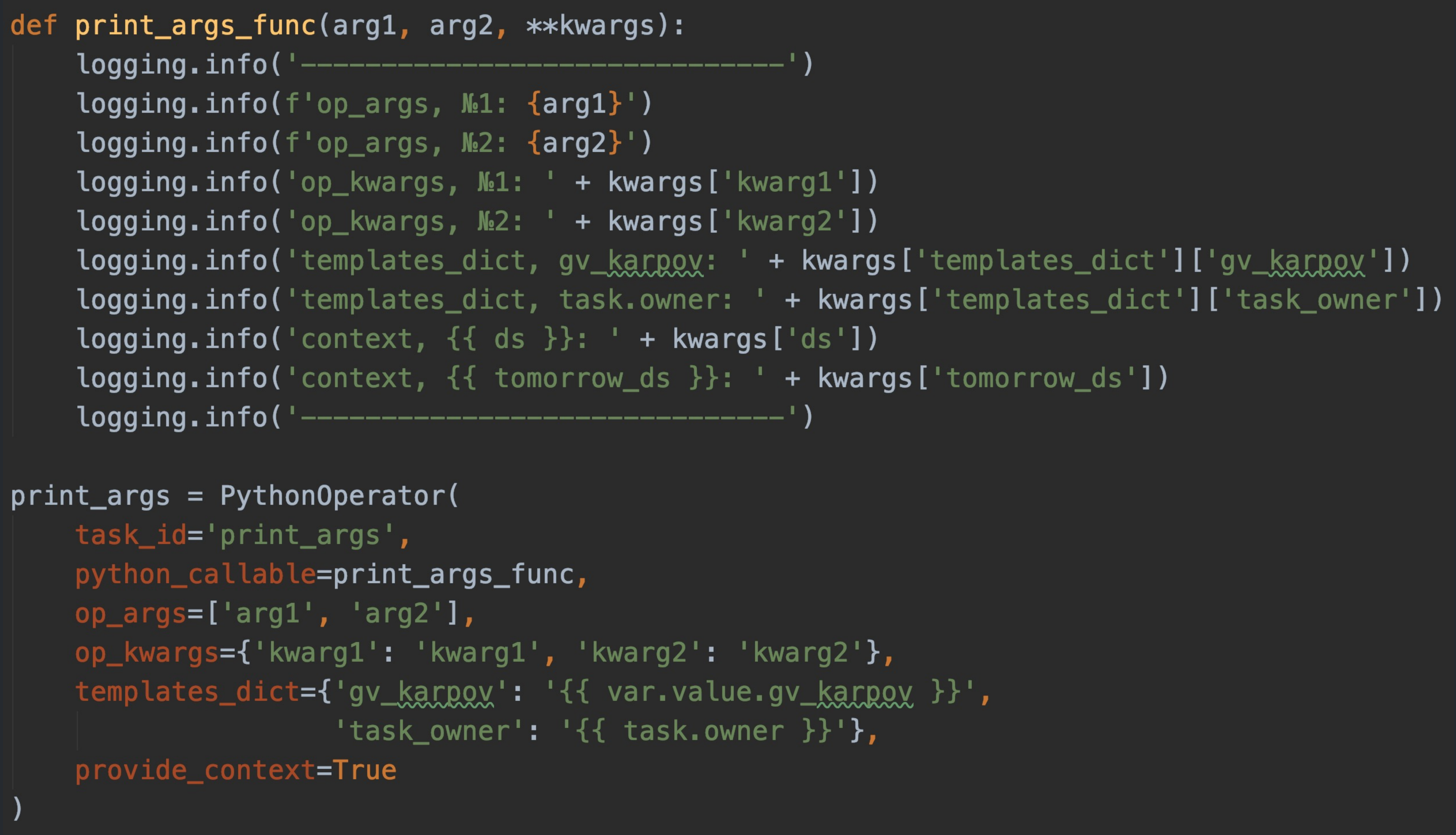


```
*** Reading local file: /var/log/airflow/dina_examples/print_templates/2021-10-08T00:00:00+00:00/8.log
[2021-10-09 22:13:05,566] {taskinstance.py:903} INFO - Dependencies all met for <TaskInstance: dina_examples.print_templates
[2021-10-09 22:13:05,605] {taskinstance.py:903} INFO - Dependencies all met for <TaskInstance: dina_examples.print_templates
[2021-10-09 22:13:05,605] {taskinstance.py:1095} INFO -
--------------------------------------------------------------------------------
[2021-10-09 22:13:05,606] {taskinstance.py:1096} INFO - Starting attempt 8 of 8
[2021-10-09 22:13:05,606] {taskinstance.py:1097} INFO -
--------------------------------------------------------------------------------
[2021-10-09 22:13:05,626] {taskinstance.py:1115} INFO - Executing <Task(PythonOperator): print_templates> on 2021-10-08T00:00
[2021-10-09 22:13:05,637] {standard_task_runner.py:52} INFO - Started process 603090 to run task
[2021-10-09 22:13:05,645] {standard_task_runner.py:76} INFO - Running: ['airflow', 'tasks', 'run', 'dina_examples', 'print_te
[2021-10-09 22:13:05,648] {standard_task_runner.py:77} INFO - Job 718: Subtask print_templates
[2021-10-09 22:13:05,752] {logging_mixin.py:109} INFO - Running <TaskInstance: dina_examples.print_templates 2021-10-08T00:00
[2021-10-09 22:13:05,932] {taskinstance.py:1254} INFO - Exporting the following env vars:
AIRFLOW_CTX_DAG_OWNER=Karpov
AIRFLOW_CTX_DAG_ID=dina_examples
AIRFLOW_CTX_TASK_ID=print_templates
AIRFLOW_CTX_EXECUTION_DATE=2021-10-08T00:00:00+00:00
AIRFLOW_CTX_DAG_RUN_ID=scheduled__2021-10-08T00:00:00+00:00
[2021-10-09 22:13:05,935] {dina_examples.py:71} INFO -
--------------------------------------------------------
ds: 2021-10-08
ds_nodash: 20211008
ts: 2021-10-08T00:00:00+00:00
gv_karpov: {'one': 1, 'two': 2}
gv_karpov, course: ETL

5 дней назад: 2021-10-03
только год: 2021
unixtime: 1633651200000
--------------------------------------------------------
[2021-10-09 22:13:05,935] {python.py:151} INFO - Done. Returned value was: None
[2021-10-09 22:13:05,955] {taskinstance.py:1219} INFO - Marking task as SUCCESS. dag_id=dina_examples, task_id=print_template
[2021-10-09 22:13:06,023] {local_task_job.py:151} INFO - Task exited with return code 0
[2021-10-09 22:13:06,072] {local_task_job.py:261} INFO - 0 downstream tasks scheduled from follow-on schedule check
```

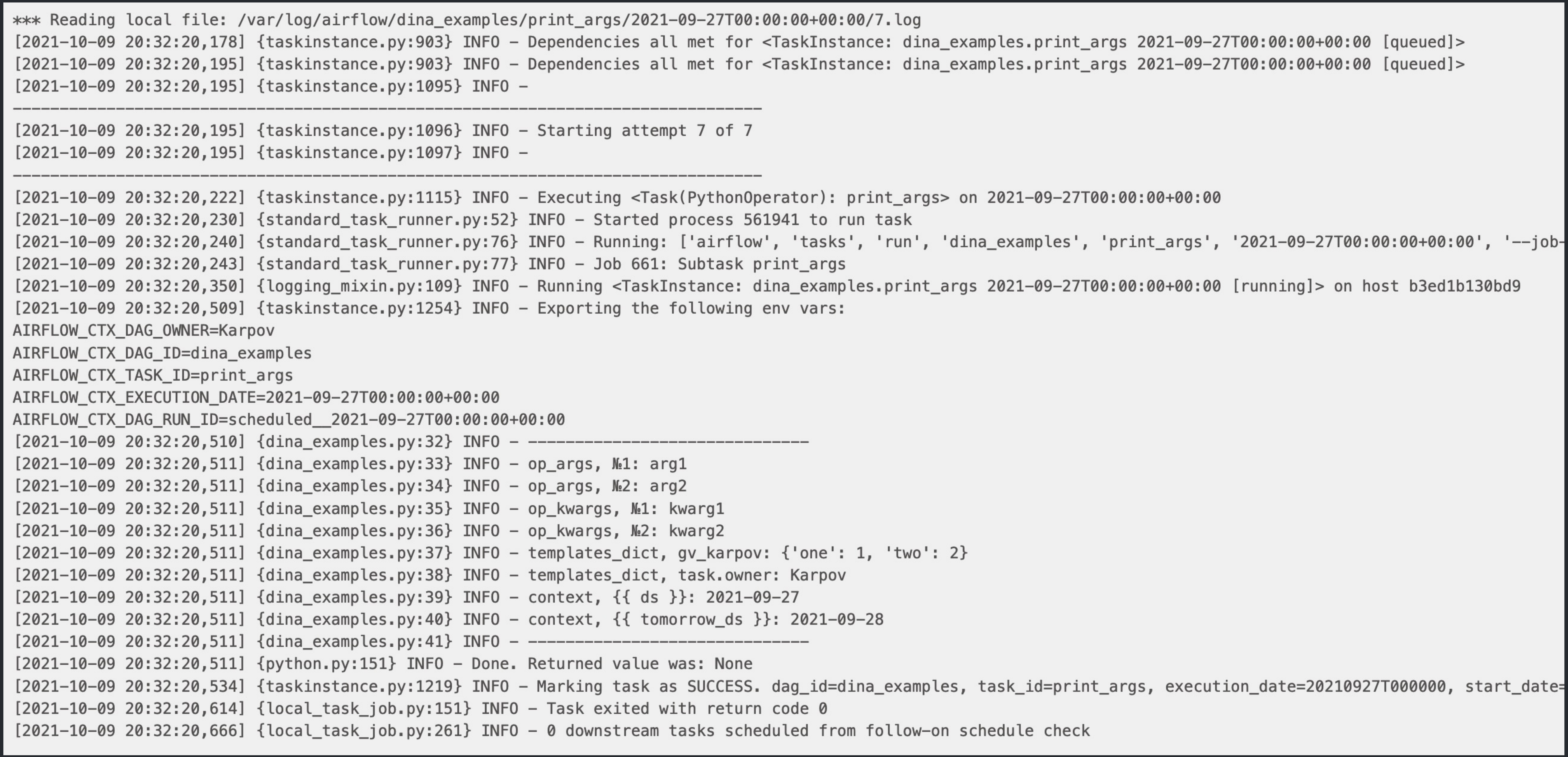
ПЕРЕДАЧА АРГУМЕНТОВ

АРГУМЕНТЫ ДЛЯ PYTHONOPERATOR

— op_args
— op_kwargs
— templates_dict
— provide_context



```python
def print_args_func(arg1, arg2, **kwargs):
    logging.info('------------------------------')
    logging.info(f'op_args, №1: {arg1}')
    logging.info(f'op_args, №2: {arg2}')
    logging.info('op_kwargs, №1: ' + kwargs['kwarg1'])
    logging.info('op_kwargs, №2: ' + kwargs['kwarg2'])
    logging.info('templates_dict, gv_karpov: ' + kwargs['templates_dict']['gv_karpov'])
    logging.info('templates_dict, task.owner: ' + kwargs['templates_dict']['task_owner'])
    logging.info('context, {{ ds }}: ' + kwargs['ds'])
    logging.info('context, {{ tomorrow_ds }}: ' + kwargs['tomorrow_ds'])
    logging.info('------------------------------')

print_args = PythonOperator(
    task_id='print_args',
    python_callable=print_args_func,
    op_args=['arg1', 'arg2'],
    op_kwargs={'kwarg1': 'kwarg1', 'kwarg2': 'kwarg2'},
    templates_dict={'gv_karpov': '{{ var.value.gv_karpov }}',
                    'task_owner': '{{ task.owner }}'},
    provide_context=True
)
```



```
*** Reading local file: /var/log/airflow/dina_examples/print_args/2021-09-27T00:00:00+00:00/7.log
[2021-10-09 20:32:20,178] {taskinstance.py:903} INFO - Dependencies all met for <TaskInstance: dina_examples.print_args 2021-09-27T00:00:00+00:00 [queued]>
[2021-10-09 20:32:20,195] {taskinstance.py:903} INFO - Dependencies all met for <TaskInstance: dina_examples.print_args 2021-09-27T00:00:00+00:00 [queued]>
[2021-10-09 20:32:20,195] {taskinstance.py:1095} INFO - 
--------------------------------------------------------------------------------
[2021-10-09 20:32:20,195] {taskinstance.py:1096} INFO - Starting attempt 7 of 7
[2021-10-09 20:32:20,195] {taskinstance.py:1097} INFO - 
--------------------------------------------------------------------------------
[2021-10-09 20:32:20,222] {taskinstance.py:1115} INFO - Executing <Task(PythonOperator): print_args> on 2021-09-27T00:00:00+00:00
[2021-10-09 20:32:20,230] {standard_task_runner.py:52} INFO - Started process 561941 to run task
[2021-10-09 20:32:20,240] {standard_task_runner.py:76} INFO - Running: ['airflow', 'tasks', 'run', 'dina_examples', 'print_args', '2021-09-27T00:00:00+00:00', '--job-
[2021-10-09 20:32:20,243] {standard_task_runner.py:77} INFO - Job 661: Subtask print_args
[2021-10-09 20:32:20,350] {logging_mixin.py:109} INFO - Running <TaskInstance: dina_examples.print_args 2021-09-27T00:00:00+00:00 [running]> on host b3ed1b130bd9
[2021-10-09 20:32:20,509] {taskinstance.py:1254} INFO - Exporting the following env vars:
AIRFLOW_CTX_DAG_OWNER=Karpov
AIRFLOW_CTX_DAG_ID=dina_examples
AIRFLOW_CTX_TASK_ID=print_args
AIRFLOW_CTX_EXECUTION_DATE=2021-09-27T00:00:00+00:00
AIRFLOW_CTX_DAG_RUN_ID=scheduled__2021-09-27T00:00:00+00:00
[2021-10-09 20:32:20,510] {dina_examples.py:32} INFO - ------------------------------
[2021-10-09 20:32:20,511] {dina_examples.py:33} INFO - op_args, №1: arg1
[2021-10-09 20:32:20,511] {dina_examples.py:34} INFO - op_args, №2: arg2
[2021-10-09 20:32:20,511] {dina_examples.py:35} INFO - op_kwargs, №1: kwarg1
[2021-10-09 20:32:20,511] {dina_examples.py:36} INFO - op_kwargs, №2: kwarg2
[2021-10-09 20:32:20,511] {dina_examples.py:37} INFO - templates_dict, gv_karpov: {'one': 1, 'two': 2}
[2021-10-09 20:32:20,511] {dina_examples.py:38} INFO - templates_dict, task.owner: Karpov
[2021-10-09 20:32:20,511] {dina_examples.py:39} INFO - context, {{ ds }}: 2021-09-27
[2021-10-09 20:32:20,511] {dina_examples.py:40} INFO - context, {{ tomorrow_ds }}: 2021-09-28
[2021-10-09 20:32:20,511] {dina_examples.py:41} INFO - ------------------------------
[2021-10-09 20:32:20,511] {python.py:151} INFO - Done. Returned value was: None
[2021-10-09 20:32:20,534] {taskinstance.py:1219} INFO - Marking task as SUCCESS. dag_id=dina_examples, task_id=print_args, execution_date=20210927T000000, start_date=
[2021-10-09 20:32:20,614] {local_task_job.py:151} INFO - Task exited with return code 0
[2021-10-09 20:32:20,666] {local_task_job.py:261} INFO - 0 downstream tasks scheduled from follow-on schedule check
```

СПАСИБО

ДИНА САФИНА