

Оглавление

Часть I ■ ПРИСТУПАЕМ К РАБОТЕ	25
1 ■ Знакомство с Apache Airflow.....	27
2 ■ Анатомия ОАГ	46
3 ■ Планирование в Airflow.....	67
4 ■ Создание шаблонов задач с использованием контекста Airflow	89
5 ■ Определение зависимостей между задачами.....	114
Часть II ■ ЗА ПРЕДЕЛАМИ ОСНОВ	144
6 ■ Запуск рабочих процессов	146
7 ■ Обмен данными с внешними системами	166
8 ■ Создание пользовательских компонентов	190
9 ■ Тестирование	222
10 ■ Запуск задач в контейнерах	259
Часть III ■ AIRFLOW НА ПРАКТИКЕ	294
11 ■ Лучшие практики.....	295
12 ■ Эксплуатация Airflow в промышленном окружении	324
13 ■ Безопасность в Airflow	369
14 ■ Проект: поиск самого быстрого способа передвижения по Нью-Йорку	393
Часть IV ■ ОБЛАКО	415
15 ■ Airflow и облако	417
16 ■ Airflow и AWS.....	426
17 ■ Airflow и Azure.....	446
18 ■ Airflow в GCP.....	465

Содержание

Предисловие	14
Благодарности	16
О книге	18
Об авторах	23
Об иллюстрации на обложке	24

Часть I ПРИСТУПАЕМ К РАБОТЕ..... 25

1 Знакомство с Apache Airflow..... 27

1.1 Знакомство с конвейерами обработки данных	28
1.1.1 Конвейеры обработки данных как графы	29
1.1.2 Выполнение графа конвейера	30
1.1.3 Графы конвейеров и последовательные сценарии	32
1.1.4 Запуск конвейера с помощью диспетчеров рабочих процессов	33
1.2 Представляем Airflow	35
1.2.1 Определение конвейеров в коде (Python) гибким образом.....	35
1.2.2 Планирование и выполнение конвейеров	36
1.2.3 Мониторинг и обработка сбоя	39
1.2.4 Инкрементальная загрузка и обратное заполнение.....	41
1.3 Когда использовать Airflow	42
1.3.1 Причины выбрать Airflow	42
1.3.2 Причины не выбирать Airflow	43
1.4 Остальная часть книги.....	44
Резюме	44

2 Анатомия ОАГ..... 46

2.1 Сбор данных из множества источников.....	46
2.1.1 Изучение данных	47
2.2 Пишем наш первый ОАГ	48
2.2.1 Задачи и операторы.....	52
2.2.2 Запуск произвольного кода на Python	53

2.3	Запуск OAG в Airflow	56
2.3.1	Запуск Airflow в окружении Python.....	56
2.3.2	Запуск Airflow в контейнерах Docker	57
2.3.3	Изучаем пользовательский интерфейс Airflow	58
2.4	Запуск через равные промежутки времени	62
2.5	Обработка неудачных задач	64
	Резюме	66
3	Планирование в Airflow	67
3.1	Пример: обработка пользовательских событий.....	68
3.2	Запуск через равные промежутки времени	69
3.2.1	Определение интервалов	70
3.2.2	Интервалы на основе Stop	71
3.2.3	Частотные интервалы	73
3.3	Инкрементная обработка данных.....	74
3.3.1	Инкрементное извлечение событий.....	74
3.3.2	Динамическая привязка ко времени с использованием дат выполнения	75
3.3.3	Разделение данных	77
3.4	Даты выполнения	80
3.4.1	Выполнение работы с фиксированными интервалами.....	80
3.5	Использование обратного заполнения.....	82
3.5.1	Назад в прошлое.....	82
3.6	Лучшие практики для проектирования задач	84
3.6.1	Атомарность.....	84
3.6.2	Идемпотентность	86
	Резюме	87
4	Создание шаблонов задач с использованием контекста Airflow	89
4.1	Проверка данных для обработки с помощью Airflow.....	90
4.1.1	Определение способа загрузки инкрементальных данных	90
4.2	Контекст задачи и шаблонизатор Jinja	92
4.2.1	Создание шаблонов аргументов оператора	92
4.2.2	Что доступно для создания шаблонов?.....	95
4.2.3	Создание шаблона для PythonOperator	97
4.2.4	Предоставление переменных PythonOperator	102
4.2.5	Изучение шаблонных аргументов.....	104
4.3	Подключение других систем	105
	Резюме	113
5	Определение зависимостей между задачами.....	114
5.1	Базовые зависимости.....	115
5.1.1	Линейные зависимости	115
5.1.2	Зависимости «один-ко-многим» и «многие-к-одному»	116
5.2	Ветвление.....	119
5.2.1	Ветвление внутри задач.....	119

5.2.2	Ветвление внутри ОАГ	121
5.3	Условные задачи	126
5.3.1	Условия в задачах	126
5.3.2	Делаем задачи условными	127
5.3.3	Использование встроенных операторов	129
5.4	Подробнее о правилах триггеров	130
5.4.1	Что такое правило триггеров?	130
5.4.2	Эффект неудач	131
5.4.3	Другие правила	132
5.5	Обмен данными между задачами	133
5.5.1	Обмен данными с помощью XCom	134
5.5.2	Когда (не) стоит использовать XCom	137
5.5.3	Использование настраиваемых XCom-бэкендов	137
5.6	Связывание задач Python с помощью Taskflow API	138
5.6.1	Упрощение задач Python с помощью Taskflow API	139
5.6.2	Когда (не) стоит использовать Taskflow API	141
	Резюме	143

Часть II ЗА ПРЕДЕЛАМИ ОСНОВ

144

6	Запуск рабочих процессов	146
6.1	Опрос условий с использованием сенсоров	147
6.1.1	Опрос пользовательских условий	150
6.1.2	Использование сенсоров в случае сбоя	152
6.2	Запуск других ОАГ	155
6.2.1	Обратное заполнение с помощью оператора TriggerDagRunOperator	159
6.2.2	Опрос состояния других ОАГ	159
6.3	Запуск рабочих процессов с помощью REST API и интерфейса командной строки	163
	Резюме	165

7	Обмен данными с внешними системами	166
7.1	Подключение к облачным сервисам	167
7.1.1	Установка дополнительных зависимостей	168
7.1.2	Разработка модели машинного обучения	169
7.1.3	Локальная разработка с использованием внешних систем	174
7.2	Перенос данных из одной системы в другую	182
7.2.1	Реализация оператора PostgresToS3Operator	184
7.2.2	Привлекаем дополнительные ресурсы для тяжелой работы	187
	Резюме	189

8	Создание пользовательских компонентов	190
8.1	Начнем с PythonOperator	191
8.1.1	Имитация API для рейтинга фильмов	191
8.1.2	Получение оценок из API	194
8.1.3	Создание фактического ОАГ	197

8.2	Создание собственного хука	199
8.2.1	Создание собственного хука	200
8.2.2	Создание ОАГ с помощью <i>MovielensHook</i>	206
8.3	Создание собственного оператора	208
8.3.1	Определение собственного оператора	208
8.3.2	Создание оператора для извлечения рейтингов	210
8.4	Создание нестандартных сенсоров	213
8.5	Упаковка компонентов	216
8.5.1	Создание пакета <i>Python</i>	217
8.5.2	Установка пакета	219
	Резюме	220

9 Тестирование

9.1	Приступаем к тестированию	223
9.1.1	Тест на благонадежность ОАГ	223
9.1.2	Настройка конвейера непрерывной интеграции и доставки	230
9.1.3	Пишем модульные тесты	232
9.1.4	Структура проекта <i>Pytest</i>	233
9.1.5	Тестирование с файлами на диске	238
9.2	Работа с ОАГ и контекстом задачи в тестах	241
9.2.1	Работа с внешними системами	246
9.3	Использование тестов для разработки	254
9.3.1	Тестирование полных ОАГ	257
9.4	Эмулируйте промышленное окружение с помощью <i>Whirl</i>	257
9.5	Создание окружений	258
	Резюме	258

10 Запуск задач в контейнерах

10.1	Проблемы, вызываемые множеством разных операторов	260
10.1.1	Интерфейсы и реализации операторов	260
10.1.2	Сложные и конфликтующие зависимости	261
10.1.3	Переход к универсальному оператору	261
10.2	Представляем контейнеры	262
10.2.1	Что такое контейнеры?	263
10.2.2	Запуск нашего первого контейнера <i>Docker</i>	264
10.2.3	Создание образа <i>Docker</i>	265
10.2.4	Сохранение данных с использованием <i>томов</i>	267
10.3	Контейнеры и <i>Airflow</i>	270
10.3.1	Задачи в контейнерах	270
10.3.2	Зачем использовать контейнеры?	270
10.4	Запуск задач в <i>Docker</i>	272
10.4.1	Знакомство с <i>DockerOperator</i>	272
10.4.2	Создание образов для задач	274
10.4.3	Создание ОАГ с задачами <i>Docker</i>	277
10.4.4	Рабочий процесс на базе <i>Docker</i>	280
10.5	Запуск задач в <i>Kubernetes</i>	281
10.5.1	Представляем <i>Kubernetes</i>	282
10.5.2	Настройка <i>Kubernetes</i>	283
10.5.3	Использование <i>KubernetesPodOperator</i>	286

10.5.4	Диагностика проблем, связанных с Kubernetes	290
10.5.5	Отличия от рабочих процессов на базе Docker	292
	Резюме	293

Часть III AIRFLOW НА ПРАКТИКЕ 294

11	Лучшие практики	295
11.1	Написание чистых ОАГ	296
11.1.1	Используйте соглашения о стилях	296
11.1.2	Централизованное управление учетными данными	300
11.1.3	Единократно указывайте детали конфигурации	301
11.1.4	Избегайте вычислений в определении ОАГ	304
11.1.5	Используйте фабричные функции для генерации распространенных шаблонов	306
11.1.6	Группируйте связанные задачи с помощью групп задач	310
11.1.7	Создавайте новые ОАГ для больших изменений	312
11.2	Проектирование воспроизводимых задач	312
11.2.1	Всегда требуйте, чтобы задачи были идемпотентными	312
11.2.2	Результаты задачи должны быть детерминированными	313
11.2.3	Проектируйте задачи с использованием парадигмы функционального программирования	313
11.3	Эффективная обработка данных	314
11.3.1	Ограничьте объем обрабатываемых данных	314
11.3.2	Инкрементальная загрузка и обработка	316
11.3.3	Кешируйте промежуточные данные	317
11.3.4	Не храните данные в локальных файловых системах	318
11.3.5	Переложите работу на внешние/исходные системы	318
11.4	Управление ресурсами	319
11.4.1	Управление параллелизмом с помощью пулов	319
11.4.2	Обнаружение задач с длительным временем выполнения с помощью соглашений об уровне предоставления услуг и оповещений	321
	Резюме	322

12	Эксплуатация Airflow в промышленном окружении	324
12.1	Архитектура Airflow	325
12.1.1	Какой исполнитель мне подходит?	327
12.1.2	Настройка базы метаданных для Airflow	328
12.1.3	Присмотримся к планировщику	330
12.2	Установка исполнителей	334
12.2.1	Настройка SequentialExecutor	335
12.2.2	Настройка LocalExecutor	335
12.2.3	Настройка CeleryExecutor	336
12.2.4	Настройка KubernetesExecutor	339
12.3	Работа с журналами всех процессов Airflow	347
12.3.1	Вывод веб-сервера	347
12.3.2	Вывод планировщика	348

12.3.3	Журналы задач	349
12.3.4	Отправка журналов в удаленное хранилище	350
12.4	Визуализация и мониторинг метрик Airflow	350
12.4.1	Сбор метрик из Airflow	351
12.4.2	Настройка Airflow для отправки метрик	353
12.4.3	Настройка Prometheus для сбора метрик	353
12.4.4	Создание дашбордов с Grafana	356
12.4.5	Что следует мониторить?	358
12.5	Как получить уведомление о невыполненной задаче	360
12.5.1	Оповещения в ОАГ и операторах	360
12.5.2	Определение соглашений об уровне предоставления услуги	362
12.6	Масштабируемость и производительность	364
12.6.1	Контроль максимального количества запущенных задач	365
12.6.2	Конфигурации производительности системы	366
12.6.3	Запуск нескольких планировщиков	367
	Резюме	368

13	Безопасность в Airflow	369
13.1	Обеспечение безопасности веб-интерфейса Airflow	370
13.1.1	Добавление пользователей в интерфейс RBAC	371
13.1.2	Настройка интерфейса RBAC	374
13.2	Шифрование хранимых данных	375
13.2.1	Создание ключа Fernet	375
13.3	Подключение к службе LDAP	377
13.3.1	Разбираемся с LDAP	378
13.3.2	Извлечение пользователей из службы LDAP	380
13.4	Шифрование трафика на веб-сервер	381
13.4.1	Разбираемся с протоколом HTTP	381
13.4.2	Настройка сертификата для HTTPS	384
13.5	Извлечение учетных данных из систем управления секретами	388
	Резюме	392

14	Проект: поиск самого быстрого способа передвижения по Нью-Йорку	393
14.1	Разбираемся с данными	396
14.1.1	Файловый ресурс Yellow Cab	397
14.1.2	REST API Citi Bike	397
14.1.3	Выбор плана подхода	399
14.2	Извлечение данных	400
14.2.1	Скачиваем данные по Citi Bike	400
14.2.2	Загрузка данных по Yellow Cab	402
14.3	Применение аналогичных преобразований к данным	405
14.4	Структурирование конвейера обработки данных	410
14.5	Разработка идемпотентных конвейеров обработки данных	411
	Резюме	414

Часть IV	ОБЛАКО	415
15	<i>Airflow и облако</i>	417
15.1	Проектирование стратегий (облачного) развертывания	418
15.2	Операторы и хуки, предназначенные для облака	420
15.3	Управляемые сервисы	421
15.3.1	<i>Astronomer.io</i>	421
15.3.2	<i>Google Cloud Composer</i>	422
15.3.3	<i>Amazon Managed Workflows for Apache Airflow</i>	423
15.4	Выбор стратегии развертывания	423
	Резюме	425
16	<i>Airflow и AWS</i>	426
16.1	Развертывание <i>Airflow</i> в <i>AWS</i>	426
16.1.1	Выбор облачных сервисов	427
16.1.2	Проектирование сети	428
16.1.3	Добавление синхронизации ОАГ	430
16.1.4	Масштабирование с помощью <i>CeleryExecutor</i>	430
16.1.5	Дальнейшие шаги	432
16.2	Хуки и операторы, предназначенные для <i>AWS</i>	432
16.3	Пример использования: бессерверное ранжирование фильмов с <i>AWS Athena</i>	434
16.3.1	Обзор	434
16.3.2	Настройка ресурсов	435
16.3.3	Создание ОАГ	438
16.3.4	Очистка	445
	Резюме	445
17	<i>Airflow и Azure</i>	446
17.1	Развертывание <i>Airflow</i> в <i>Azure</i>	446
17.1.1	Выбор сервисов	447
17.1.2	Проектирование сети	448
17.1.3	Масштабирование с помощью <i>CeleryExecutor</i>	449
17.1.4	Дальнейшие шаги	450
17.2	Хуки и операторы, предназначенные для <i>Azure</i>	451
17.3	Пример: бессерверное ранжирование фильмов с <i>Azure Synapse</i>	452
17.3.1	Обзор	452
17.3.2	Настройка ресурсов	453
17.3.3	Создание ОАГ	457
17.3.4	Очистка	463
	Резюме	464
18	<i>Airflow в GCP</i>	465
18.1	Развертывание <i>Airflow</i> в <i>GCP</i>	465
18.1.1	Выбор сервисов	466
18.1.2	Развертывание в <i>GKE</i> с помощью <i>Helm</i>	468

18.1.3	Интеграция с сервисами Google.....	471
18.1.4	Проектирование сети.....	472
18.1.5	Масштабирование с помощью CeleryExecutor	473
18.2	Хуки и операторы, предназначенные для GCP	476
18.3	Пример использования: бессерверный рейтинг фильмов в GCP.....	481
18.3.1	Загрузка в GCS.....	481
18.3.2	Загрузка данных в BigQuery.....	483
18.3.3	Извлечение рейтингов, находящихся в топе	485
	Резюме	488
Приложение А	Запуск примеров кода	490
Приложение В	Структуры пакетов Airflow 1 и 2	494
Приложение С	Сопоставление метрик в Prometheus	498
	Предметный указатель	500

Предисловие

Нам обоим посчастливилось работать инженерами по обработке данных в интересные и трудные времена.

Хорошо это или плохо, но многие компании и организации осознают, что данные играют ключевую роль в управлении их операциями и их улучшении. Последние разработки в области машинного обучения и искусственного интеллекта открыли множество новых возможностей для извлечения выгоды.

Однако внедрение процессов, ориентированных на данные, часто бывает затруднительным, поскольку обычно требуется координировать работу в различных гетерогенных системах и связывать все воедино аккуратно и своевременно для последующего анализа или развертывания продукта.

В 2014 году инженеры Airbnb осознали проблемы управления сложными рабочими процессами данных внутри компании. Чтобы разрешить их, они приступили к разработке Airflow: решения с открытым исходным кодом, позволяющего писать и планировать рабочие процессы, а также отслеживать их выполнение с помощью встроеного веб-интерфейса.

Успех этого проекта быстро привел к тому, что его приняли в рамках Apache Software Foundation, сначала в качестве проекта инкубатора в 2016 году, а затем в качестве проекта верхнего уровня в 2019 году. В результате многие крупные компании теперь используют Airflow для управления многочисленными критически важными процессами обработки данных.

Работая консультантами в GoDataDriven, мы помогли клиентам внедрить Airflow в качестве ключевого компонента в проектах, связанных с созданием озер и платформ данных, моделей машинного обучения и т. д. При этом мы поняли, что передача этих решений может быть непростой задачей, поскольку такие сложные инструменты, как Airflow, трудно освоить в одночасье. По этой причине мы также разработали программу по обучению Airflow в GoData-Driven, часто

организовывали семинары и участвовали в них, чтобы поделиться своими знаниями, мнениями и даже пакетами с открытым исходным кодом. В совокупности эти усилия помогли нам изучить тонкости работы с Airflow, которые не всегда было легко понять, используя доступную нам документацию.

В этой книге мы хотим предоставить исчерпывающее введение в Airflow, которое охватывает все, от построения простых рабочих процессов до разработки собственных компонентов и проектирования / управления развертываниями Airflow. Мы намерены дополнить блоги и другую онлайн-документацию, объединив несколько тем в одном месте в кратком и понятном формате. Тем самым мы надеемся придать вам стимул в работе с Airflow, опираясь на опыт, который мы накопили, и преодолевая трудности, с которыми мы столкнулись за последние годы.

Благодарности

Выход этой книги был бы невозможен без поддержки множества замечательных людей. Коллеги из GoDataDriven и наши друзья поддерживали нас и предоставили ценные предложения и важные наблюдения. Кроме того, читатели, члены программы Manning Early Access Program (MEAP), оставили полезные комментарии на онлайн-форуме.

Рецензенты, участвовавшие в процессе подготовки издания, также предоставили полезные отзывы: Эл Кринкер, Клиффорд Тербер, Дэниел Ламблин, Дэвид Криф, Эрик Платон, Фелипе Ортега, Джейсон Рендель, Джереми Чен, Джасти Пик, Джонатан Вуд, Картик Сирасанагандла, Кент Р. Спиллнер, Лин Чен, Филип Бест, Филип Паттерсон, Рамбабу Поза, Ричард Мейнсен, Роберт Г. Гимбель, Роман Павлов, Сальваторе Кампанья, Себастьян Пальма Мардонес, Торстен Вебер, Урсин Стаусс и Влад Навицкий.

Мы выражаем особую благодарность Брайану Союеру из издательства Manning, редактору отдела закупок, который помог нам сформировать первоначальную заявку на издание этой книги и поверил, что мы сможем довести дело до конца; Трише Лувар, нашему редактору-консультанту по аудитории, которая очень терпеливо отвечала на все наши вопросы, давала важные отзывы по каждой из наших черновики глав и была для нас важным проводником на протяжении всего пути; а также остальным сотрудникам: редактору проекта Дейдре Хиама; редактору Мишель Митчелл; корректору Кери Хейлз и техническому корректору Элу Кринкеру.

Бас Харенслак

Я хотел бы поблагодарить своих друзей и семью за их терпение и поддержку в течение этого приключения, длившегося полтора года, которое превратилось из второстепенного проекта в бесчисленное количество дней, ночей и выходных. Стефани, спасибо за то, что все время терпела меня, пока я работал за компьютером. Мириам, Герд и Лотте,

спасибо за то, что терпели меня и верили в меня, пока я писал эту книгу. Я также хотел бы поблагодарить команду GoDataDriven за их поддержку и стремление всегда учиться и совершенствоваться. Пять лет назад я и представить себе не мог, когда начал работать, что стану автором книги.

Джулиан де Руйтер

Прежде всего я хотел бы поблагодарить свою жену Анн Полин и сына Декстера за их бесконечное терпение в течение многих часов, которые я потратил на то, чтобы «еще немного поработать» над книгой. Эта книга была бы невозможна без их непоколебимой поддержки. Также хотел бы поблагодарить нашу семью и друзей за их поддержку и доверие. Наконец, хочу сказать спасибо нашим коллегам из GoDataDriven за их советы и поддержку, от которых я также многому научился за последние годы.

О книге

Эта книга была написана, чтобы помочь вам реализовать рабочие процессы (или конвейеры), ориентированные на обработку данных с помощью Airflow. Она начинается с концепций и механики, участвующих в программном построении рабочих процессов для Apache Airflow с использованием языка программирования Python. Затем мы переходим к более подробным темам, таким как расширение Airflow путем создания собственных компонентов и всестороннего тестирования рабочих процессов. Заключительная часть книги посвящена проектированию и управлению развертывания Airflow, затрагивая такие темы, как безопасность и проектирование архитектур для облачных платформ.

Кому адресована эта книга

Эта книга написана для специалистов и инженеров по обработке данных, которые хотят разрабатывать базовые рабочие процессы в Airflow, а также для инженеров, интересующихся более сложными темами, такими как создание собственных компонентов для Airflow или управление развертываниями Airflow. Поскольку рабочие процессы и компоненты Airflow построены на языке Python, мы ожидаем, что у читателей имеется промежуточный опыт программирования на Python (т. е. они хорошо умеют создавать функции и классы Python, имеют представление о таких понятиях, как `* args` и `** kwargs` и т. д.). Также будет полезен опыт работы с Docker, поскольку большинство примеров кода запускаются с использованием Docker (хотя при желании их можно запустить локально).

Структура книги

Книга состоит из четырех разделов, которые охватывают 18 глав.

Часть I посвящена основам Airflow. В ней объясняется, что такое Airflow, и изложены его основные концепции:

- в главе 1 обсуждается концепция рабочих процессов / конвейеров обработки данных и их создание с помощью Apache Airflow. В ней также рассказывается о преимуществах и недостатках Airflow по сравнению с другими решениями, в том числе приводятся ситуации, при которых вы, возможно, не захотите использовать Apache Airflow;
- глава 2 посвящена базовой структуре конвейеров в Apache Airflow (также известных как OAG), с объяснением различных задействованных компонентов и их совместимости;
- в главе 3 показано, как использовать Airflow для планирования работы конвейеров через повторяющиеся промежутки времени, чтобы можно было (например) создавать конвейеры, которые постепенно загружают новые данные с течением времени. В этой главе также рассматриваются тонкости механизма планирования Airflow, который часто является источником путаницы;
- в главе 4 показано, как использовать механизмы создания шаблонов в Airflow для динамического включения переменных в определения конвейера. Это позволяет ссылаться на такие вещи, как даты выполнения расписания в конвейерах;
- в главе 5 демонстрируются различные подходы к определению отношений между задачами в конвейерах, что позволяет создавать более сложные структуры конвейеров с ветками, условными задачами и общими переменными.

В части II подробно рассматривается использование более сложных тем, включая взаимодействие с внешними системами, создание собственных компонентов и разработку тестов для ваших конвейеров:

- в главе 6 показано, как запускать рабочие процессы другими способами, не связанными с фиксированным расписанием, такими как загрузка файлов или вызовы по протоколу HTTP;
- в главе 7 демонстрируются рабочие процессы с использованием операторов, которые управляют различными задачами вне Airflow, что позволяет создавать поток событий через системы, которые не связаны между собой;
- в главе 8 объясняется, как создавать собственные компоненты для Airflow, которые позволяют повторно использовать функциональные возможности в разных конвейерах или интегрироваться с системами, не поддерживающими встроенные функции Airflow;
- в главе 9 обсуждаются различные варианты тестирования рабочих процессов Airflow, затрагиваются свойства операторов и способы их применения во время тестирования;

- в главе 10 показано, как использовать рабочие процессы на базе контейнеров для выполнения задач конвейера в Docker или Kubernetes, а также обсуждаются преимущества и недостатки этих подходов.

Часть III посвящена применению Airflow на практике и затрагивает такие темы, как передовые методы, запуск и обеспечение безопасности Airflow и последний демонстрационный пример:

- в главе 11 рассказывается о передовых методах построения конвейеров, которые помогут вам разрабатывать и внедрять эффективные и удобные в сопровождении решения;
- в главе 12 подробно описано несколько тем, которые необходимо учитывать при запуске Airflow в промышленном окружении, например архитектуры для горизонтального масштабирования, мониторинга, журналирования и оповещений;
- в главе 13 обсуждается, как обезопасить установку Airflow, чтобы избежать нежелательного доступа и минимизировать воздействие в случае взлома;
- в главе 14 показан пример проекта Airflow, в котором мы периодически обрабатываем поездки с использованием сервисов Yellow Cab и Citi Bikes по Нью-Йорку, чтобы определить самый быстрый способ передвижения между районами.

В части IV исследуется, как запускать Airflow в облачных платформах. Она включает такие темы, как проектирование развертываний Airflow для облачных платформ и использование встроенных операторов для взаимодействия с облачными сервисами:

- в главе 15 дается общее введение, в котором рассказывается, какие компоненты Airflow участвуют в (облачных) развертываниях, представлена идея, лежащая в основе облачных компонентов, встроенных в Airflow, и рассматриваются варианты самостоятельной реализации развертывания в облаке в сравнении с использованием управляемого решения;
- глава 16 посвящена облачной платформе Amazon AWS. Здесь рассказывается о решениях для развертывания Airflow на AWS и демонстрируется, как применять определенные компоненты для использования сервисов AWS;
- в главе 17 разрабатываются развертывания и демонстрируются облачные компоненты для платформы Microsoft Azure;
- в главе 18 рассматриваются развертывания и облачные компоненты для платформы Google GCP.

Тем, кто плохо знаком с Airflow, следует прочитать главы 1 и 2, чтобы получить внятное представление о том, что такое Airflow, и его возможностях. В главах 3–5 представлена важная информация об основных функциях Airflow. В остальной части книги обсуждаются такие темы, как создание собственных компонентов, тестирование, передовые практики и развертывание, и ее можно читать в произвольном порядке в зависимости от конкретных потребностей читателя.

О коде

Весь исходный код в листингах или тексте набран моноширинным шрифтом, чтобы отделить его от обычного текста. Иногда также используется **жирный шрифт**, чтобы выделить код, который изменился по сравнению с предыдущими шагами в главе, например когда к существующей строке кода добавляется новая функция.

Во многих случаях оригинальный исходный код был переформатирован; мы добавили разрывы строк и переработали отступы, чтобы уместить их по ширине книжных страниц. В редких случаях, когда этого оказалось недостаточно, в листинги были добавлены символы продолжения строки (`\>`). Кроме того, комментарии в исходном коде часто удаляются из листингов, когда описание кода приводится в тексте. Некоторые листинги сопровождаются аннотациями, выделяющие важные понятия.

Ссылки на элементы в коде, сценарии или определенные классы, переменные и значения Airflow часто выделяются *курсивом*, чтобы их было легче отличить от окружающего текста.

Исходный код всех примеров и инструкции по их запуску с помощью Docker и Docker Compose доступны в нашем репозитории GitHub (<https://github.com/BasPH/data-pipelines-with-apache-airflow>), а также на сайте издательства «ДМК Пресс» www.dmkpress.com на странице с описанием соответствующей книги.

ПРИМЕЧАНИЕ В приложении А представлены более подробные инструкции по запуску примеров кода.

Все примеры кода были протестированы с использованием Airflow 2.0. Большинство примеров также следует запускать в более старых версиях Airflow (1.10) с небольшими изменениями. Там, где это возможно, мы включили встроенные указатели, как это сделать. Чтобы вам было легче учитывать различия в путях импорта между Airflow 2.0 и 1.10, в приложении В представлен обзор измененных путей импорта.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге, – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Manning Publications очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Об авторах

БАС ХАРЕНСЛАК – инженер по обработке данных в GoDataDriven, компании, занимающейся разработкой решений на основе данных, расположенной в Амстердаме, Нидерланды. Имея опыт разработки программного обеспечения и информатики, он любит работать над программным обеспечением и данными, как если бы они были сложными головоломками. Он предпочитает работать над ПО с открытым исходным кодом, участвует в проекте Apache Airflow и является одним из организаторов семинара Amsterdam Airflow.

ДЖУЛИАН ДЕ РУИТЕР – инженер машинного обучения, имеет опыт работы в области компьютерных наук и наук о жизни, а также имеет докторскую степень в области вычислительной биологии рака. Будучи опытным разработчиком программного обеспечения, он любит соединять миры науки о данных и инженерии, используя облачное программное обеспечение и программное обеспечение с открытым исходным кодом для разработки решений машинного обучения, готовых к промышленной эксплуатации. В свободное время он с удовольствием разрабатывает собственные пакеты Python, участвует в проектах с открытым исходным кодом и возится с электроникой.

Об иллюстрации на обложке

Рисунок на обложке называется «Femme de l'Isle de Siphanto», или *Женщина с острова Сифанто*. Иллюстрация взята из коллекции костюмов разных стран Жака Грассе де Сен-Совера (1757–1810), «Costumes de Différents Pays», изданной во Франции в 1797 году.

Каждая иллюстрация нарисована и раскрашена вручную. Богатое разнообразие коллекции Грассе де Сен-Совера наглядно напоминает нам о том, насколько обособленными в культурном отношении были города и регионы мира всего 200 лет назад. Изолированные друг от друга люди говорили на разных диалектах и языках. На улице или в деревне было легко определить, где они живут и чем занимаются или каково их положение в обществе, по их одежде.

С тех пор наша манера одеваться изменилась, а разнообразие регионов, столь богатых в то время, исчезло. Сейчас трудно отличить жителей разных континентов, не говоря уже о разных городах, регионах или странах. Возможно, если смотреть на это оптимистично, мы обменяли культурное и визуальное разнообразие на более разнообразную частную жизнь или более разнообразную интеллектуальную и техническую жизнь.

Мы в Manning высоко ценим изобретательность, инициативу и, конечно, радость от компьютерного бизнеса с книжными обложками, основанными на разнообразии жизни в разных регионах два века назад, которое оживает благодаря рисункам Грассе де Сен-Совера.

Конец ознакомительного фрагмента.

Приобрести книгу можно

в интернет-магазине

«Электронный универс»

e-Univers.ru