4.1 Упаковка десктопного приложения

Упаковка десктопного приложения

Приложение Flet на Python и все его зависимости могут быть упакованы в исполняемый файл, и пользователь может запустить его на своем компьютере без установки интерпретатора Python или каких-либо модулей.

Flet использует API PyInstaller для упаковки приложения Flet на Python и всех его зависимостей в один пакет для Windows, macOS и Linux. Чтобы создать пакет для Windows, PyInstaller должен быть запущен на Windows; чтобы создать приложение для Linux, он должен быть запущен на Linux; и чтобы создать приложение для macOS

- на macOS.

Начните с установки PyInstaller:

pip install pyinstaller

Перейдите в каталог, где находится ваш файл .py , и соберите ваше приложение с помощью следующей команды:

flet pack your\_program.py # ваша\_программа

Теперь ваше приложение Flet должно быть доступно в папке dist . Попробуйте запустить программу, чтобы увидеть, работает ли она.

**На macOS:**

open dist/your\_program.app

**на Windows:**

dist\your\_program.exe

**на Linux:**

dist/your\_program

Теперь вы можете просто заархивировать содержимое папки и

распространять его среди своих пользователей! Им не нужно устанавливать Python или Flet для запуска вашей упакованной программы - отличная альтернатива Electron!

dist

По умолчанию исполняемый файл/пакет имеет то же имя, что и скрипт на Python. Вы можете изменить его с помощью аргумента --name :

flet pack your\_program.py --name bundle\_name # имя\_пакета

# Настройка иконки пакета

По умолчанию иконка приложения-пакета - это дискета, что может запутать молодых разработчиков, пропустивших те древние времена, когда для хранения компьютерных данных использовались дискеты.

Вы можете заменить иконку на свою, добавив аргумент --icon :

flet pack your\_program.py --icon <your\_icon.png>

PyInstaller преобразует предоставленный PNG в формат, специфичный для

платформы ( .ico для Windows и установить модуль Pillow:

.icns

для macOS), но для этого вам нужно

pip install pillow

# Упаковка ресурсов

Ваше приложение Flet может включать ресурсы (assets). Если предоставленные

ресурсы приложения находятся в папке assets рядом с , они

ваша\_программа.py

могут быть добавлены в пакет приложения с помощью аргумента --add-data , на macOS/Linux:

assets;assets

flet pack your\_program.py --add-data "assets:assets"

На Windows должны быть разделены символом ; :

flet pack your\_program.py --add-data "assets;assets"

# Настройка пакета macOS

Детали пакета macOS могут быть настроены с помощью следующих специфических для macOS аргументов flet pack :

* - отображаемое имя пакета macOS, показываемое в Dock, Activity Monitor, диалоге "О программе".

--product-name

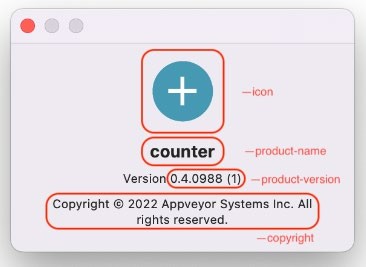
--product-version

* - версия пакета, показываемая в диалоге "О программе".
* - уведомление об авторских правах, показываемое в диалоге "О программе".

--copyright

* - уникальный ID пакета.

--bundle-id



# Настройка метаданных исполняемого файла Windows

Свойства "Details" исполняемого файла Windows можгут быть настроены с помощью следующих аргументов flet pack :

* - поле "Название продукта".

--product-name

* --product-version - поле "Версия продукта".
* - поле "Версия файла".

--file-version

* --file-description - поле "Описание файла", также отображаемое имя программы в диспетчере задач.

--copyright

* - поле "Авторские права".

# Использование Неприрывной Интеграции (CI) для упаковки на нескольких платформах

Чтобы создать пакет приложения с помощью PyInstaller для конкретной ОС, он должен быть запущен на этой ОС.

Если у вас нет доступа к Mac или PC, вы можете собрать свое приложение для всех трех платформ с помощью AppVeyor - сервиса непрерывной интеграции для Windows, Linux и macOS.

Если вкратце, то непрерывная интеграция (CI) - это автоматизированный процесс сборки, тестирования и развертывания (непрерывная доставка - CD) приложения при каждом пуше в репозиторий.

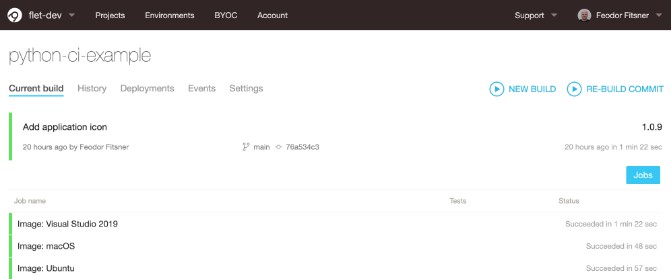
AppVeyor бесплатен для открытых проектов, размещенных на GitHub, GitLab и Bitbucket. Чтобы использовать AppVeyor, отправьте свое приложение в репозиторий одного из этих провайдеров управления исходным кодом.

*ПРИМЕЧАНИЕ*

*Вы можете просто форкнуть репозиторий и настроить его под свои потребности.*

*flet-dev/python-ci-example*

Когда вы отправляете любые изменения в репозиторий GitHub, AppVeyor автоматически запускает новую сборку:



Вот что делает этот поток CI при каждом пуше в репозиторий:

* Клонирует репозиторий на чистую виртуальную машину.
* Устанавливает зависимости приложения с помощью pip .
* Запускает macOS и Ubuntu.

flet pack

для упаковки приложения Python в пакет для Windows,

* Архивирует пакеты приложений и загружает их в "Artifacts".
* Загружает пакеты приложений в релизы GitHub, когда новый тег отправляется. Просто отправьте новый тег, чтобы сделать релиз!

**GITHUB\_TOKEN**

в - это токен личного доступа GitHub (PAT или

GITHUB\_TOKEN

appveyor.yml

Personal Access Token), который использует AppVeyor для публикации созданных пакетов в релизы репозитория. Вам нужно сгенерировать свой собственный токен и заменить его в appveyor.yml .

Войдите в свою учетную запись GitHub и перейдите на страницу личного токена доступа. Нажмите "Generate new token" и выберите область видимости (scope) "public\_repo" или "repo" для общедоступного или приватного репозитория соответственно. Скопируйте сгенерированный токен в буфер обмена и вернитесь в портал AppVeyor. Перейдите на страницу *Encrypt configuration data* и вставьте токен в поле "Value to encrypt", затем нажмите кнопку "Encrypt". Поместите зашифрованное

значение под в вашем appveyor.yml .

GITHUB\_TOKEN

Настройте AppVeyor для вашего проекта на Python, отправьте новый тег в репозиторий и "автоматически" получите десктопный пакет для всех трех платформ в релизах GitHub! 🎉🎉

В дополнение к релизам GitHub, вы также можете настроить выпуск артефактов в корзину Amazon S3 или Azure Blob Storage.

4.1 Упаковка десктопного приложения