2.2 Использование клиентского хранилища в Flet

Использование клиентского хранилища в Flet

API клиентского хранилища Flet позволяет сохранять пары ключ-значение на стороне клиента в постоянном хранилище. Реализация Flet использует

пакет в Flutter.

shared\_preferences

Конкретный механизм хранения зависит от платформы, на которой работает приложение Flet:

* Web - Локальное хранилище.
* Desktop - JSON файл.
* iOS - NSUserDefaults.
* Android - SharedPreferences. Запись данных в хранилище: **строки**

page.client\_storage.set("ключ", "значение")

## числа, булевы значения

page.client\_storage.set("настройка.числа", 12345)

page.client\_storage.set("булева\_настройка", True)

**списки**

page.client\_storage.set("любимые\_цвета", ["красный", "зеленый", "синий"])

### ПРИМЕЧАНИЕ

*Каждое приложение Flutter, использующее плагин shared\_preferences , имеет свой собственный набор настроек. Поскольку один и тот же клиент Flet (который является приложением Flutter) используется для запуска пользовательского*

*интерфейса для нескольких приложений Flet, любые значения, сохраненные в одном*

*приложении Flet, видны/доступны для другого приложения Flet, запущенного тем же пользователем.*

*Чтобы отличить настройки одного приложения от другого, рекомендуется использовать уникальный префикс для всех ключей хранилища,*

*например {компания}.{продукт} .. Например, чтобы сохранить токен аутентификации в одном приложении, вы можете использовать ключ acme.первое\_приложение.токен\_аутентификации, а в другом приложении*

*acme.second\_app.auth\_token*

*использовать ентификации).*

*(acme.второе\_приложение.токен\_аут*

### ВНИМАНИЕ

*Ответственностью разработчика приложения Flet является шифрование конфиденциальных данных перед их отправкой в клиентское хранилище, чтобы они не были прочитаны/изменены другим приложением или пользователем приложения.*

## Чтение данных:

Значение автоматически преобразуется обратно в оригинальный тип

value = page.client\_storage.get("ключ")

colors = page.client\_storage.get("любимые\_цвета")

# colors = ["красный", "зеленый", "синий"]

Проверить наличие ключа:

page.client\_storage.contains\_key("ключ") # Возвращает True, если ключ существует

Получить все ключи:

page.client\_storage.get\_keys("префикс-ключа.")

Удалить значение:

page.client\_storage.remove("ключ")

Очистить хранилище:

page.client\_storage.clear()

## ВНИМАНИЕ!

*Функция*

*clear()*

*- это опасная функция, которая удаляет все настройки всех*

*приложений Flet, когда-либо запущенных тем же пользователем, и служит предупреждением о том, что постоянные данные приложения не должны храниться в клиентском хранилище.*

В этом уроке мы рассмотрели основы работы с клиентским хранилищем Flet. Мы научились записывать, читать и удалять данные, а также получать все ключи и проверять наличие ключа. Помните, что важно использовать уникальный префикс для всех ключей хранилища, чтобы отличать настройки одного приложения от другого. Также не забывайте о необходимости шифрования конфиденциальных данных перед их отправкой в клиентское хранилище.