

# Документация к приложению «Агрегатор погоды (10 источников)»

## 1. Введение

**Агрегатор погоды** — настольное приложение на языке Python с графическим интерфейсом (Tkinter), предназначенное для сбора, отображения и усреднения метеорологических данных из нескольких источников. Приложение объединяет данные от российских и международных погодных сервисов, отображает их в удобной таблице и вычисляет средние значения основных метеопараметров.

Приложение ориентировано на:

- учебные и демонстрационные цели;
- анализ расхождений прогнозов разных источников;
- практику работы с парсингом, многопоточностью и GUI.

## 2. Назначение и возможности приложения

### 2.1 Назначение

Программа предназначена для автоматизированного получения погодных данных по выбранному городу и их наглядного представления пользователю.

### 2.2 Основные возможности

- получение данных о погоде из **10 различных источников**;
- поддержка нескольких городов (Москва, Санкт-Петербург, Новосибирск, Екатеринбург, Казань);
- параллельный сбор данных с использованием потоков;
- автоматическая генерация реалистичных данных при ошибках парсинга;
- расчет средних значений температуры, влажности, давления и скорости ветра;
- сохранение результатов в файл формата JSON;
- ведение истории запусков;
- логирование всех операций в интерфейсе.

## 3. Используемые технологии и библиотеки

### 3.1 Язык программирования

- **Python 3.x**

### 3.2 Основные библиотеки

- **tkinter** — графический пользовательский интерфейс;
- **ttk** — стилизованные элементы интерфейса;
- **requests** — HTTP-запросы к погодным сайтам;
- **BeautifulSoup (bs4)** — парсинг HTML-страниц;
- **threading** — многопоточность;
- **queue** — безопасный обмен данными между потоками;
- **dataclasses** — структурированное хранение данных;
- **json** — сохранение данных и истории;
- **re** — обработка текстовых данных;
- **datetime** — работа с датой и временем.

## 4. Архитектура приложения

Приложение построено по модульному принципу и включает следующие ключевые компоненты:

- **WeatherData** — модель данных погоды;
- **WeatherScraper** — набор методов для получения и генерации данных;
- **WeatherApp** — основной класс GUI и логики приложения;
- **main()** — точка входа в программу.

## 5. Описание структуры данных

### 5.1 Класс WeatherData

Класс WeatherData используется для хранения данных, полученных из одного источника.

**Поля класса:**

- source — название источника;
- temperature — температура воздуха (°C);
- feels\_like — ощущаемая температура (°C);
- humidity — влажность воздуха (%);
- pressure — атмосферное давление (мм рт. ст.);
- wind\_speed — скорость ветра (м/с);
- description — текстовое описание погоды;
- timestamp — время получения данных.

## 6. Получение данных о погоде

### 6.1 Класс WeatherScraper

Класс WeatherScraper содержит статические методы для получения данных о погоде.

Источники данных:

1. Gismeteo.ru
2. Яндекс.Погода
3. Sinoptik.ua
4. Pogoda.mail.ru
5. Meteoinfo.ru
6. Foreca.ru
7. Meteoweb.ru
8. Rp5.ru
9. Weather.com
10. BBC Weather

Для части источников используется **реальный HTML-парсинг**, для остальных — **генерация приближенных значений**, имитирующих реальные данные.

## 6.2 Обработка ошибок

Если получение данных с сайта невозможно:

- приложение фиксирует предупреждение в логе;
- автоматически генерируются правдоподобные значения;
- источник помечается как «сгенерированный» или «ошибка».

# 7. Графический интерфейс пользователя

## 7.1 Основные элементы интерфейса

- выпадающий список выбора города;
- кнопки управления:
  - «Обновить данные»;
  - «Сохранить в JSON»;
  - «Очистить всё»;
- таблица с данными от источников;
- панель средних значений;
- окно логирования;
- строка состояния с текущим временем.

## 7.2 Таблица данных

В таблице отображаются следующие параметры:

- источник данных;
- температура;
- ощущаемая температура;
- влажность;
- давление;
- скорость ветра;
- время обновления;
- статус данных.

Цветовая индикация строк:

- зелёный — реальные данные;
- жёлтый — сгенерированные данные;
- красный — данные после ошибки.

## 8. Многопоточность и производительность

Сбор данных осуществляется в отдельном потоке, что позволяет:

- не блокировать графический интерфейс;
- отображать прогресс выполнения;
- безопасно обновлять UI через очередь сообщений (Queue).

## 9. Расчет и отображение средних значений

После получения данных выполняется расчет средних значений:

- температуры;
- ощущаемой температуры;
- влажности;
- давления;
- скорости ветра.

Результаты отображаются в отдельной панели с цветовой индикацией температуры.

## 10. Сохранение данных

### 10.1 Экспорт в JSON

Пользователь может сохранить результаты текущего запроса в файл формата JSON. Файл содержит:

- название города;
- дату и время сохранения;
- количество источников;

- данные каждого источника;
- рассчитанные средние значения.

## 10.2 История запусков

Приложение автоматически сохраняет краткую информацию о каждом запуске в файл истории. Хранится не более 50 последних записей.

## 11. Логирование

Все ключевые действия приложения отображаются в лог-окне:

- начало и завершение сбора данных;
- успешные операции;
- предупреждения;
- ошибки.

Лог содержит временные метки и цветовую индикацию уровней сообщений.

## 12. Требования к запуску

### 12.1 Системные требования

- Python версии 3.8 и выше;
- доступ в интернет (для реального парсинга);
- ОС Windows / Linux / macOS.

### 12.2 Установка зависимостей

```
pip install requests beautifulsoup4
```

## 13. Заключение

Приложение «Агрегатор погоды» демонстрирует комплексный подход к разработке настольных программ на Python, объединяя GUI, сетевые запросы,

парсинг данных, многопоточность и работу с файлами. Архитектура приложения позволяет легко расширять список источников и функциональность в будущем.