BlooTube Voice API

Версия интерфейса **v1** · Приложение **1.1.14** · Редакция **2026-10-19**.

API создаёт задания озвучки и возвращает готовые MP3. Доступ предоставляется на тарифах START, PRO и ULTRA. Во время технического обслуживания сервер может возвращать `503 SERVICE_UNAVAILABLE`.

Доступ и ключи

Базовый адрес: **https://api.apexcorelink.org/voice/api/v1**.

Тариф	Доступ к API
FREE	Недоступен, включая ранее выданные ключи
START	1 активный ключ
PRO STUDIO	2 активных ключа
ULTRA VIP	До 3 активных ключей

Публичный REST API — отдельный интерфейс, а не полный набор всех экранов Mini App. Через него доступны методы, перечисленные в справочнике ниже: профиль и лимиты, каталог голосов, задания озвучки, история заданий и аудио, предпрослушивание, клонирование и управление API-ключами. Избранное, лента новостей, заявки на тариф и административная панель остаются функциями Mini App. Для нескольких текстов отправляйте отдельное задание на каждый текст; основные параметры речи передаются в теле запроса.

Создайте ключ в Mini App: **Кабинет** → **API для разработчиков**. Секрет показывается один раз. Храните его на своём сервере или в переменной окружения; не размещайте в клиентском JavaScript, публичном репозитории или ссылке.

Каждый запрос, кроме `GET /capabilities`, требует заголовок:

```
Authorization: Bearer YOUR_BTV_API_KEY
```

Допускается `X-API-Key` вместо `Authorization`. Все ключи одного аккаунта видят общие задания и используют общую квоту с Mini App. Отзыв ключа немедленно прекращает доступ. После окончания или понижения тарифа лишние ключи приостанавливаются; повышение тарифа не включает их автоматически.

Первая озвучка

1. Получите профиль и текущие лимиты: `GET /me`.
2. Выберите `voice_id` из `GET /voices?language=ru&limit=1`.
3. Отправьте JSON в `POST /tasks` с новым `Idempotency-Key`.
4. Из ответа **202** прочитайте `task.task_id`.
5. Проверяйте `GET /tasks/{task_id}` примерно раз в две секунды.
6. При `completed` скачайте `task.download_url` с тем же заголовком авторизации.

```
{
  "text": "Привет! Это проверка озвучки.",
  "title": "Проверка озвучки",
  "voice_id": "VOICE_ID_FROM_CATALOG",
  "language": "ru",
  "pause_durations_ms": {"sentence": 200, "paragraph": 400, "chapter": 600}
}
```

Ответ о постановке в очередь содержит вложенный объект `task`, а не готовое аудио:

```
{
  "ok": true,
  "replayed": false,
  "message": "Task queued successfully",
  "task": {
    "task_id": "a591f67a-ae1e-4d44-8952-8378096cb46a",
    "title": "Проверка озвучки",
    "filename": "Проверка озвучки.mp3",
    "status": "queued",
    "text_length": 29,
    "chars_reserved": 29,
    "total_chunks": 1,
    "created_at": "2026-09-20T00:00:00+00:00"
  }
}
```


Полные запускаемые примеры: Python, Node.js, Bash/cURL. Python использует только стандартную библиотеку; для Bash нужны `curl` (7.76+) и `python3`. Перед запуском задайте `BTV_API_KEY` в окружении. Не вводите ключ в текст руководства. Примеры печатают ID операции, выбранный голос и ID задания, но не секрет. Для повторного запуска после потери ответа сохраните `BTV_REQUEST_ID` и `BTV_VOICE_ID`, не меняя текст. Python и Node.js повторяют временные ошибки до 30 минут на запрос; окно настраивается через `BTV_REQUEST_WAIT_SECONDS`. Bash использует повторы cURL (до 100, в пределах 30 минут). После исчерпания окна сохраните ID и продолжите позднее.

Скачивание возвращает `audio/mpeg`, не JSON. Всегда проверяйте HTTP-статус и тип содержимого до сохранения файла.

Квота, очередь и повторы

`GET /me` и `GET /usage` возвращают `quota.used_chars`, `reserved_chars`, `remaining_chars`, `quota_day` и `resets_at`. Точные действующие лимиты аккаунта находятся в `GET /me → limits`; не фиксируйте цены или числовые квоты в интеграции.

Суточная квота обновляется в **00:00 UTC**. Все ключи и Mini App делят один баланс. Символ - кодовая точка Unicode исходного текста: считаются пробелы, переводы строк и разметка. Например, один обычный emoji считается одним символом; составной emoji может состоять из нескольких. Дополнительные внутренние теги при разбиении не списываются повторно.

При принятии задания объём резервируется. Задание, перешедшее через полночь, остаётся привязанным к исходному `quota_day`. После завершения резерв освобождается и фиксируется фактическое списание в `chars_committed`. `task.chars_reserved` сохраняет первоначально зарезервированный объём для истории; текущий общий резерв находится в `quota.reserved_chars`. Проверяйте итоговый `chars_committed` после ошибки или отмены, а не вычисляйте возврат по числу частей.

`Idempotency-Key` - строка из 1–128 видимых ASCII-символов без пробелов, например UUID. Одинаковое тело с тем же ключом возвращает прежнюю задачу и `replayed: true`, включая уже завершённую. Другое тело с тем же ключом даёт **409**. Ключ действует в пределах аккаунта, а не одного API-ключа. Без него каждый повтор создаёт новое задание.

Вместо заголовка можно передать `client_task_id`. При передаче обоих значения должны совпасть. После таймаута создания повторяйте исходное тело с прежним ключом. После таймаута опроса продолжайте опрашивать прежний `task_id`.

Лимиты частоты разделены для создания, опроса и остальных операций. Mini App и все API-ключи расходуют общий лимит аккаунта; ключ и IP также имеют защитные ограничения. Фактические значения находятся в `GET /me → limits.rate_limits`. При **429**, **503** или **507** выдержите `Retry-After`, если он указан; заголовок может содержать секунды либо дату HTTP. При временных **500/502/504** и сетевом таймауте повторяйте с увеличением паузы до 30 секунд. Для повторного `POST /tasks` обязательно сохраняйте исходное тело и тот же `Idempotency-Key`; для опроса — прежний `task_id`. Ошибки авторизации, параметров и квоты не следует повторять бесконечно. Streaming и webhooks в API v1 отсутствуют.

Несколько озвучек одновременно

Отправляйте каждый текст отдельным `POST /tasks` со своим `Idempotency-Key`, собственным `voice_id` и, при необходимости, `title`. Запросы можно отправлять параллельно; каждый возвращает собственный `task_id`, статус, MP3 и расход символов. В Mini App голос выбирается отдельно в карточке каждого текста. Кнопка внутри карточки запускает только этот текст, нижняя кнопка отправляет все открытые карточки, которые ещё не выполняются. Общие настройки фиксируются для каждого задания в момент запуска.

`GET /me → limits.max_parallel_jobs` — максимальное число разных заданий, обрабатываемых одновременно. `in_flight_concurrency` — максимальное число одновременно обрабатываемых частей **суммарно по аккаунту**, а не на каждое задание. `max_queued_jobs` включает и ожидающие, и выполняемые задания; `max_backlog_chunks` ограничивает их незавершённые части. Индивидуально допустимые задания помещаются в выделенные слоты очереди, если хватает общей квоты и ресурсов сервиса.

Тариф	Частей одновременно, по умолчанию	Заданий одновременно, по умолчанию	Заданий в работе и очереди, по умолчанию
FREE	1	1	1
START	5	3	5
PRO	10	5	10
ULTRA	15	10	25

Администратор может менять значения. Всегда используйте текущие лимиты из `GET /me`; для FREE они доступны в Mini App, публичный API требует START, PRO или ULTRA. Очередь и ограничения общие для Mini App и всех API-ключей аккаунта. Увеличение числа ключей не увеличивает тарифные лимиты. Свободная мощность распределяется по аккаунтам, а внутри аккаунта — между его заданиями. Максимум тарифа не гарантирует выделенную мощность при любой общей нагрузке: при занятости выполнение замедляется, а принятые задания ожидают своей очереди.

Временная занятость синтеза, пауза по `Retry-After` или нехватка места для сборки не требуют создавать новое задание. Сервер сохраняет готовые части и продолжает обработку; общий срок выполнения принятого задания — до 24 часов. Невосстановимая ошибка параметров или исчерпание ограниченного бюджета запросов с неизвестным результатом могут завершить его ошибкой; проверяйте итоговый статус и `chars_committed`. Отказ **429/503/507** при создании не резервирует символы и не выдаёт новый `task_id`. Mini App автоматически повторяет временно отклонённую отправку, пока открыта, и восстанавливает сохранённое задание при следующем открытии.

Для отдельных заданий достаточно опроса раз в 3–5 секунд; для пачки можно получать список `GET /tasks` и сопоставлять сохранённые `task_id`. Учитывайте пагинацию и общий `limits.rate_limits.poll_tasks_rpm`. При **429** приостанавливайте опрос всего аккаунта на указанное время. Истечение таймаута вашего клиента не отменяет серверное задание.

Суммарный объём всех отправленных текстов должен укладываться в общий остаток символов. При повторе после таймаута сохраняйте для каждого текста исходный ключ и тело запроса. Менять название с тем же ключом нельзя — это новое тело и ответ 409.

Голоса и параметры речи

Каталог: `GET /voices`. Фильтры: `language`, `gender` (`female`, `male`, `neutral`), `category` (`catalog`, `cloned`, `my_clones`) и поиск `q`. Пагинация: `limit` 1–1000, по умолчанию 100; `offset` от 0. Ответ содержит `total`, `limit`, `offset`, `voices`.

`GET /voices/{voice_id}` возвращает доступные `accents` и `locales`. `language`/`languages` в каталоге описывают голос, а `language` задания - язык вашего текста. Голос может говорить на другом поддерживаемом языке. Акцент по умолчанию подбирается для языка; если нужного варианта нет, возможно произношение с исходным акцентом голоса. Явный `accent` должен быть в его списке. Он может намеренно отличаться от языка текста.

Поле	Описание
`text` (обязательно)	Исходный текст и разметка BlooTube Voice. До 1 200 000 кодовых точек Unicode; действующий лимит аккаунта — GET /me → limits.max_job_chars.
`voice_id` (обязательно)	Идентификатор каталожного, общего или собственного личного голоса.
`title`	Название результата без .mp3; например, имя загруженного .txt. Возвращается в истории и используется как имя MP3 при скачивании. По умолчанию: `null`.
`language`	Базовый язык синтеза (ISO 639-1, например 'ru', 'en'). Взаимоисключающий с параметром 'locale'. По умолчанию: `null`.
`locale`	Региональная языковая подача / вариант (BCP-47, например 'en-US', 'es-MX'). Код должен быть в GET /capabilities; акцент проверяется отдельно по выбранному голосу. Взаимоисключающий с параметром 'language'. По умолчанию: `null`.
`accent`	Опциональный акцент речи (например, 'british', 'mexican'). Строго валидируется по возможностям конкретного голоса. По умолчанию: `null`.
`normalization`	Режим нормализации текста: 'auto' (раскрытие чисел, сумм, дат), 'off' (отключить автоматическую нормализацию) или код локали BCP-47. По умолчанию: `"auto"`.
`speed`	Скорость генерации речи от 0.6 до 1.5. По умолчанию: `1.0`.
`volume`	Громкость генерации речи от 0.5 до 2.0. По умолчанию: `1.0`.
`dsp_speed`	Постобработка темпа без изменения высоты тона от 1.0 до 2.0. По умолчанию: `1.0`.
`pause_mode`	Умные паузы: none — выключены; sentence — после конца предложения; paragraph — между блоками через пустую строку; chapter — вместо отдельной строки ---. По умолчанию: `"none"`.
`pause_duration_ms`	Длительность умной паузы от 100 до 5000 миллисекунд. Используется, когда pause_mode не равен none; явный <break> имеет приоритет. По умолчанию: `1000`.
`pause_durations_ms`	Независимые паузы в миллисекундах: sentence, paragraph и/или chapter, каждое значение от 100 до 5000 с шагом 100. Отсутствующий ключ отключает соответствующую паузу. Если поле передано, оно заменяет pause_mode и pause_duration_ms; пустой объект отключает все умные паузы. По умолчанию: `null`.
`emotion`	Эмоциональная подсказка из GET /capabilities → emotions. Базовые пресеты доступны в emotion_presets. Экспериментальный эффект зависит от голоса, языка и смысла текста; естественная выразительность Sonic 3.6 не требует тегов. По умолчанию: `null`.
`generation_config`	Совместимые настройки Cartesia: speed, volume, emotion. Применяются к этому запросу. Противоречащие плоские параметры отклоняются; неизвестные поля запрещены. По умолчанию: `null`.
`client_task_id`	Необязательный ID операции; правила как для Idempotency-Key. По умолчанию: `null`.

Если `language` и `locale` отсутствуют, используется основной язык голоса (или `ru`, если он неизвестен). Рекомендуется всегда задавать язык текста. Неизвестные поля JSON отвергаются.

Полные допустимые языки, локали, эмоции и диапазоны: `GET /capabilities`. `normalization: off` отключает автоматическое раскрытие чисел и дат, но не гарантирует чтение каждого символа буквально. Для чтения по буквам используйте `<spell>`.

`pause_durations_ms` позволяет включать паузы независимо: `sentence` — после конца предложения, `paragraph` — между блоками через пустую строку, `chapter` — вместо отдельной строки `---` либо `***`. Укажите только нужные

ключи и длительность каждого от 100 до 5000 мс с шагом 100 мс. Пустой объект отключает все умные паузы. На совпадающих границах приоритет у главы, затем у абзаца, затем у предложения; паузы не складываются. Маркер главы не озвучивается, а явный `<break>` имеет приоритет и не дублируется. Для старых клиентов сохраняются `pause_mode` и `pause_duration_ms` с одной общей паузой; если передано `pause_durations_ms`, оно заменяет старые поля. Настройки входят в идентичность запроса: при повторе с тем же `Idempotency-Key` значения должны оставаться неизменными.

Предпрослушивание: `POST /voices/{voice_id}/preview` возвращает 202 и `state`. Повторяйте этот же запрос до `ready`, затем открывайте `preview_url`. Подготовка ограничена шестью новыми примерами в минуту и 60 за 24 часа на аккаунт. Обычное открытие URL примера само не запускает озвучку.

Разметка текста

Разметка применяется внутри поля `text`. Доступны два варианта управляющих тегов: парные области `speed`, `volume`, `emotion` либо самозакрывающиеся управляющие теги Cartesia. `spell` всегда парный, `break` самозакрывающийся. Незакрытые и неподдерживаемые теги дают 422; не смешивайте два способа управления в одном тексте.

```
Обычный текст. <speed ratio="0.8">Медленнее.</speed>
<volume ratio="1.2">Громче.</volume>
Пауза <break time="1s"/> и <spell>API</spell>.
```

Подсказки эмоций экспериментальны: официальная документация ограничивает гарантированную работу emotion-тегов английским; естественная выразительность Sonic 3.6 определяется контекстом и голосом во всех поддерживаемых языках. Для русского проверяйте результат короткой пробой. Пауза: от `100ms` до `10s`. Маркер `[laughter]` добавляет смех. Изменения произношения зависят от голоса и текста; проверяйте короткий образец перед длинной озвучкой.

Клонирование голоса

Создание клона - отдельная асинхронная операция. `POST /clones` принимает **multipart/form-data**, поле аудио называется **file**, а `Idempotency-Key` обязателен. Не путайте `clone_id` операции с `voice_id` готового голоса.

Поле	Значение
`file`	Речь 5–30 секунд, файл от 10 КБ до 16 МБ; WAV, MP3, OGG, FLAC, M4A, AAC или WebM
`name`	Название, 1–64 символа
`language`	Поддерживаемый язык, по умолчанию `ru`
`gender`	`female` или `male`, по умолчанию `female`
`description`	Необязательно, до 256 символов
`shared_in_app`	Необязательно: `true` по умолчанию — доступен всем в BlooTube Voice; `false` — только владельцу в Mini App и API BlooTube Voice
`public_voice_consent`	`true`: согласие на передачу образца Cartesia и создание публичного голоса
`voice_rights_confirmed`	`true`: подтверждение прав на образец

При `shared_in_app=false` клон остаётся доступен владельцу в BlooTube Voice, но скрыт от других аккаунтов в каталоге и недоступен им по `voice_id` для озвучки и предпрослушивания. Настройка действует только в BlooTube Voice: создание голоса в Cartesia остаётся публичным, и настройка не ограничивает доступ по ID через Cartesia. Передавайте подтверждения только после явного согласия владельца образца. Клонирование чужого голоса без соответствующих прав не допускается.

Пример Bash/cURL (укажите свой файл и название; флаги согласия допустимы только после получения согласия):

```
curl --fail-with-body --silent --show-error \
  -H "Authorization: Bearer $BTV_API_KEY" \
  -H "Idempotency-Key: $CLONE_REQUEST_ID" \
  -F "file=@sample.wav" -F "name=Мой голос" \
  -F "language=ru" -F "gender=female" \
  -F "public_voice_consent=true" -F "voice_rights_confirmed=true" \
  "https://api.apexcorelink.org/voice/api/v1/clones"
```

Заранее сохраните уникальный `CLONE_REQUEST_ID`, например UUID. Повтор с тем же ID должен содержать тот же файл и поля. Чтобы создать личный клон в BlooTube Voice, добавьте `-F "shared_in_app=false"`. При смене этого параметра для новой операции используйте новый `Idempotency-Key`.

В ответе 202 получите `clone.clone_id`. Опрос `GET /clones` показывает состояния `queued → preparing → creating → ready`. Использовать в озвучке можно только `voice_id` клона в состоянии `ready`. `created_at` клона передаётся в секундах Unix UTC. Для личного клона `GET /voices/{voice_id}/preview.mp3` требует заголовок `Authorization: Bearer <API-ключ владельца>`; публичные примеры по-прежнему открываются без него.

`failed` означает подтверждённый сбой. `unknown` означает потерянный ответ: не создавайте клон заново с другим ключом; передайте `clone_id` поддержке для сверки. Эти состояния различаются, чтобы не создавать дубликаты и не терять занятый слот. За последние 24 часа допускается до трёх новых операций на каждый слот клонирования (минимум три); повтор с тем же ID новой операцией не считается. Число слотов смотрите в `GET /me → limits.max_clone_slots`.

`DELETE /clones/{voice_id}` также принимает `clone_id` и возвращает 202. При `delete_pending`/`deleting` голос уже скрыт из каталога; слот освобождается после подтверждения удаления. При `delete_failed` можно повторить удаление. Чужие клоны удалять нельзя; голос с незавершёнными заданиями даёт 409.

Статусы, аудио и ошибки

Статус задания	Что делать
`queued`	Задание принято и ожидает обработки
`processing`	Продолжайте опрос; `progress` показывает процент готовых частей (0–100), а не оставшееся время
`completed`	Проверьте `file_state` и скачайте `download_url` до `expires_at`
`failed`	Опрос закончен; прочитайте `error`, сохраните `task_id` для поддержки
`cancelled`	Опрос закончен; проверьте итоговый расход `chars_committed`

`POST /tasks/{task_id}/cancel` повторяем. Выполняющийся запрос может сначала завершиться; ориентируйтесь на возвращаемый фактический статус. Если итоговый файл не создан из-за ошибки или отмены, весь резерв возвращается: `chars_committed` равен 0. Готовые внутренние части отдельно не выдаются. Уже завершённое задание отменить нельзя; истечение срока хранения готового файла не возвращает потраченную квоту.

Если незавершённый синтез достигает общего срока 24 часа с момента принятия, задание получает `failed` с кодом `JOB_DEADLINE_EXCEEDED`. Уже выполняемый запрос сначала завершается, затем резерв возвращается. Ожидание мощности само по себе не вызывает отдельную ошибку после 30 минут. Актуальные пределы доступны в `capabilities.task_timeouts`; `no_progress_seconds: 0` означает, что отдельный таймаут отсутствия прогресса отключён. Если все части уже готовы, сборка MP3 может завершиться после общего срока. После окончательной ошибки и устранения причины создайте новое задание с новым `Idempotency-Key`.

Файлы обычно хранятся 5 часов после завершения. Источником истины служит `expires_at`: срок может настраиваться оператором. `GET /tasks/{id}/download` и `HEAD` требуют действующий ключ владельца аккаунта. После истечения срока возвращается 410. Поддержка диапазонов `Range` позволяет продолжить загрузку. Сохраняйте нужное аудио у себя до истечения срока.

Необязательное `title` (до 120 символов, без разделителей пути) задаёт читаемое имя результата. Например, для `Сказка о царе Маяги.txt` передайте `"title": "Сказка о царе Маяги"`: в задании вернётся `filename: "Сказка о царе Маяги.mp3"`, скачивание и отправка в Telegram используют это имя. Без `title` именем файла служит ID

задания. Название не меняет голос, текст или учёт символов.

Список заданий: `GET /tasks?limit=20&offset=0`. Максимум 100 записей за запрос; фильтры `status` и `api_key_id`. Ключи одного аккаунта могут читать, скачивать и отменять общие задания. `file_state`: `pending` — файл ещё готовится, `available` — доступен, `unavailable` — не создан, `expired` — срок истёк, `deleted` — удалён владельцем через Mini App. Только доступный файл получает `download_url`; относительный URL разрешайте от домена API.

Ошибка API имеет единый формат:

```
{"error":{"code":"RATE_LIMIT_EXCEEDED","message":"Too many
requests.","request_id":"req_1234567890abcdef"}}
```

HTTP	Действие
400 / 422	Исправьте тело, параметры или разметку; тот же запрос не повторяйте
401	Проверьте ключ и заголовок авторизации
402	Не хватает доступной квоты с учётом резерва
403	Проверьте лицензию, приостановку ключа и доступ аккаунта
404	Проверьте ID и принадлежность объекта аккаунту
409	Конфликт параметров, повторного ключа операции или текущего состояния
410	Аудио больше недоступно
413	Уменьшите размер запроса или файла
429	Выдержите `Retry-After`, уменьшите частоту/очередь
500 / 502 / 503 / 507	Временная ошибка сервиса или нехватка места; сохраните ID и повторите позже

`X-Request-ID` возвращается в ответах приложения. Сообщайте поддержке этот ID и `task_id`/`clone_id`, а не секретный ключ или полный текст озвучки. При ошибке HTTPS-прокси ответ может быть HTML; клиент должен проверять тип содержимого перед разбором JSON.

Справочник и файлы для разработчиков

OpenAPI - машиночитаемая схема методов, полей и ответов. Она нужна для импорта в Postman и генерации клиента; для обычного чтения достаточно этого руководства.

Скачать: схему JSON, схему YAML, Markdown. Актуальная веб-версия.

Метод	Назначение
`GET /me`	Профиль, лицензия и лимиты
`GET /capabilities`	Поддерживаемые параметры
`GET /voices`	Каталог с фильтрами и пагинацией
`GET /voices/{voice_id}`	Данные голоса и акценты
`GET /usage`	Расход и остаток квоты
`POST /tts`	Создать задание озвучки (совместимый устаревающий псевдоним `/tasks`)
`POST /tasks`	Создать задание озвучки
`GET /tasks`	Список заданий аккаунта

Метод	Назначение
`GET /tasks/{task_id}`	Статус и результат задания
`HEAD /tasks/{task_id}/download`	Скачать MP3 или получить заголовки
`GET /tasks/{task_id}/download`	Скачать MP3 или получить заголовки
`POST /tasks/{task_id}/cancel`	Отменить незавершённое задание
`POST /voices/{voice_id}/preview`	Подготовить пример голоса
`POST /clones`	Создать публичный клон
`GET /clones`	Свои клоны и операции
`DELETE /clones/{voice_id}`	Удалить свой клон

Административные методы и управление ключами через Telegram не входят в Bearer API. Версия v1 допускает добавление полей: клиентам следует игнорировать неизвестные поля. Изменения несовместимого контракта должны выходить в новой основной версии.

Редакция 1.1.14: актуальные лимиты аккаунта и число одновременных заданий возвращает `GET /me`. Поддержка: @bloo_kz.

Пример Python целиком

```
"""Python 3.10+, standard library only. Set BTV_API_KEY before running."""
import json
import os
from email.utils import parsedate_to_datetime
from http.client import IncompleteRead
from pathlib import Path
import time
from urllib.error import HTTPError, URLError
from urllib.parse import urljoin, urlsplit
from urllib.request import Request, HTTPRedirectHandler, build_opener
import uuid

BASE = os.getenv("BTV_BASE_URL", "https://api.apexcorelink.org/voice/api/v1").rstrip("/")
KEY = os.environ["BTV_API_KEY"]
REQUEST_ID = os.getenv("BTV_REQUEST_ID") or str(uuid.uuid4())
OUTPUT = Path(os.getenv("BTV_OUTPUT", "voice.mp3"))


class NoRedirect(HTTPRedirectHandler):
    def redirect_request(self, req, fp, code, msg, headers, newurl):
        return None  # Never forward credentials through a redirect.


opener = build_opener(NoRedirect())


def request(method, path, body=None, *, operation_id=None, audio=False):
    relative = path if path.startswith("/voice/") else path.lstrip("/")
    url = urljoin(BASE + "/", relative)
    origin = lambda value: (urlsplit(value).scheme, urlsplit(value).netloc)

    if origin(url) != origin(BASE):
        raise RuntimeError("Refusing to send the key to a different host")
    headers = {"Authorization": "Bearer " + KEY}
    if operation_id: headers["Idempotency-Key"] = operation_id
    data = json.dumps(body, ensure_ascii=False).encode() if body is not None else None
    if data is not None: headers["Content-Type"] = "application/json"
    deadline = time.monotonic() + float(os.getenv("BTV_REQUEST_WAIT_SECONDS", "1800"))
    attempt = 0
```

```
    while True:
        try:
            with opener.open(Request(url, data, headers, method=method), timeout=30) as response:
                content = response.read()
                if audio:
                    if response.headers.get_content_type() != "audio/mpeg" or not content:
                        raise RuntimeError("Expected an MP3 download")
                    return content
                return json.loads(content)
        except HTTPError as error:
            if (error.code not in (408, 429, 500, 502, 503, 504, 507) or
                    time.monotonic() >= deadline or (method == "POST" and not operation_id)):
                try:
                    details = json.loads(error.read()).get("error", {})
                except (ValueError, AttributeError):
                    details = {}
                message = f"HTTP {error.code}: {details.get('code')}"
                message += f"; request_id={details.get('request_id')}"
                raise RuntimeError(message) from None
            try:
                value = error.headers.get("Retry-After", "3")
                delay = max(1, float(value))

            except ValueError:
                try: delay = max(1, parsedate_to_datetime(value).timestamp() - time.time())
                except (TypeError, ValueError, OverflowError): delay = 3
        except (TimeoutError, URLError, ConnectionError, IncompleteRead):
            if time.monotonic() >= deadline or (method == "POST" and not operation_id): raise
            delay = min(30, 2 ** min(attempt, 5))
        if time.monotonic() + delay > deadline:
            raise TimeoutError("Retry window ended; keep the same Idempotency-Key or resume polling
this ID")
        attempt += 1
        time.sleep(delay)


def main():
    # Save this ID: after a timeout, repeat the same body with this BTV_REQUEST_ID.
    print("Idempotency-Key:", REQUEST_ID)
    request("GET", "me")
    voices = request("GET", "voices?category=catalog&language=ru&limit=1")["voices"]
    if not voices: raise RuntimeError("No suitable voice found")
    voice_id = os.getenv("BTV_VOICE_ID") or voices[0]["id"]
    print("BTV_VOICE_ID:", voice_id)
    receipt = request("POST", "tasks", {
        "text": "Привет! Это проверка API BlooTube Voice.",
        "voice_id": voice_id, "language": "ru",
    }, operation_id=REQUEST_ID)
    task_id = receipt["task"]["task_id"]
    print("Task:", task_id)
    deadline = time.monotonic() + float(os.getenv("BTV_POLL_WAIT_SECONDS", "3600"))
    while time.monotonic() < deadline:
        task = request("GET", "tasks/" + task_id)["task"]

        if task["status"] == "completed":
            audio = request("GET", task["download_url"], audio=True)
            temporary = OUTPUT.with_suffix(OUTPUT.suffix + ".tmp")
            temporary.write_bytes(audio)
            temporary.replace(OUTPUT)
            print("Saved:", OUTPUT)
            return
        if task["status"] in ("failed", "cancelled"):
            raise RuntimeError(f"Task {task_id}: {task['status']}; {task.get('error')}")
        time.sleep(2)
    raise TimeoutError(f"Task {task_id} is still running; resume polling this ID")


if __name__ == "__main__": main()
```

Совместимость выразительной озвучки

Основой подготовки текста служит актуальная документация Cartesia Sonic 3.6. Пишите естественные законченные реплики с обычным регистром и пунктуацией. Не используйте ЗАГЛАВНЫЕ БУКВЫ как команду кричать. Эмоция должна соответствовать смыслу; при важной смене подачи лучше отправлять отдельные фразы. Полный принимаемый перечень находится в `GET /capabilities` → `emotions`, а семь привычных пресетов сохранены в `emotion_presets`.

Поддерживается официальный самозакрывающийся синтаксис:

```
<emotion value="surprised"/><speed ratio="1.1"/>Неужели получилось!
<emotion value="scared"/><speed ratio="0.9"/>Я... я слышу шаги.
<emotion value="content"/>Вот это история! [laughter]
```

Такая настройка действует до следующего тега того же вида, в том числе при внутреннем разбиении длинного текста. Старые парные теги продолжают работать. В одном тексте выбирайте один способ управления: парные области либо самозакрывающиеся настройки. Смешивание этих двух способов отклоняется с 422; `<break>` и `<spell>` доступны в обоих вариантах. Неподдерживаемые названия эмоций, некорректные числа, NaN и бесконечность отклоняются до резервирования квоты.

Вместо плоских `speed`, `volume`, `emotion` можно передавать `generation_config` с этими же полями. Противоречащие значения отклоняются. Указанные одним способом настройки имеют ту же идентичность запроса, что и эквивалентные плоские настройки. Полный REST-протокол Cartesia, включая потоковые endpoints, не становится автоматически доступен через этот адаптер: здесь остаются задания BlooTube и готовый MP3.

Нативный невербальный маркер: `[laughter]`. Теги `[gasp]`, `[whisper]`, `[cry]` и `[tremble]` не заявлены как поддерживаемые команды. Запинки и эмоциональные междометия задавайте словами в самой реплике. Дрожание, если оно добавляется звуковой обработкой, следует отличать от возможностей синтезатора.

Источники: https://docs.cartesia.ai/build-with-cartesia/tts-models/latest и https://docs.cartesia.ai/build-with-cartesia/capability-guides/ssml-tags и https://docs.cartesia.ai/build-with-cartesia/capability-guides/prompting-tips . Проверено 19 октября 2026.