

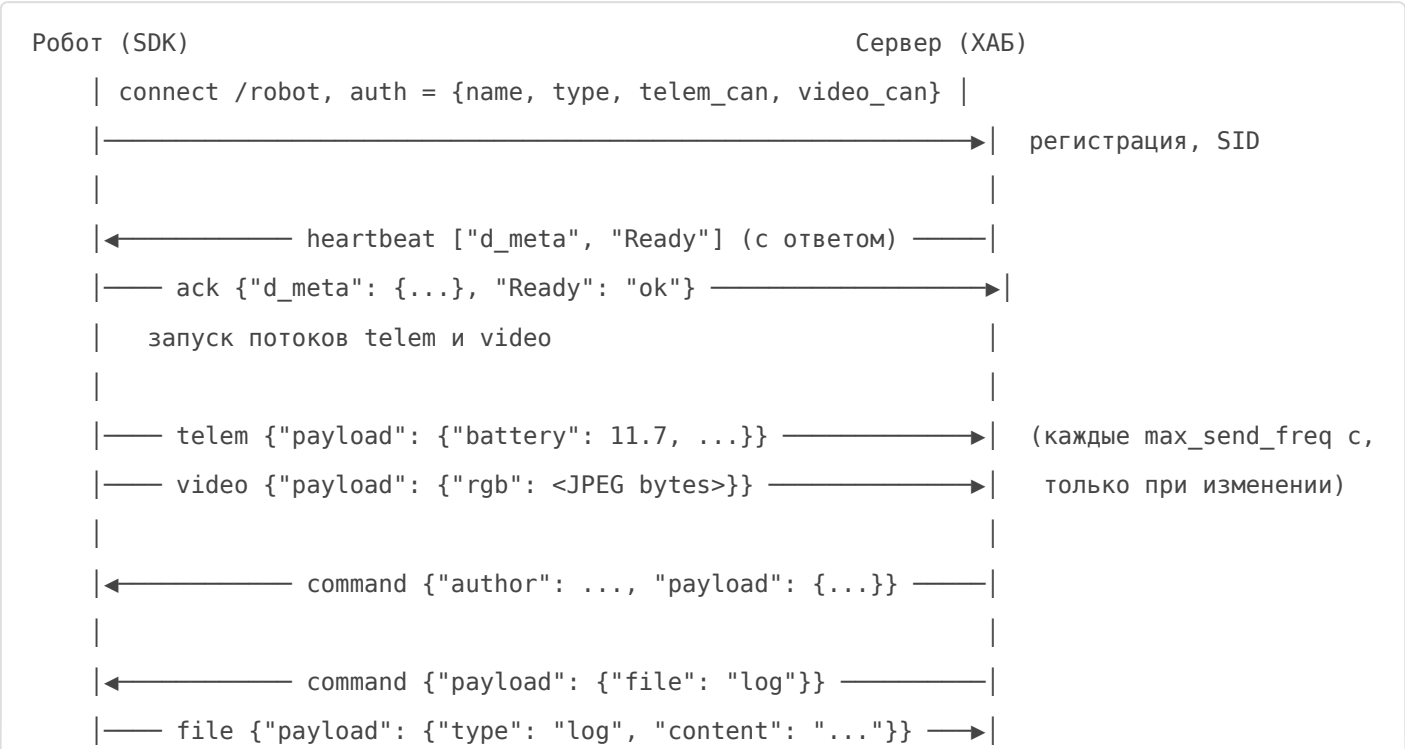
????????? ??????? ? ???????????

Описание протокола обмена между SDK `artex-robot` (версия 0.1.1) и сервером ARTEX. Страница полезна тем, кто реализует собственного клиента (на другом языке, для микроконтроллера или симулятора) или серверную часть.

??????????

Параметр	Значение
Протокол	Socket.IO (Engine.IO поверх WebSocket / HTTP long-polling)
Адрес	<code><server>:<port></code> , по умолчанию <code>http://artex.arcon:80</code>
Namespace	<code>/robot</code>
Авторизация	Объект <code>auth</code> при подключении = паспорт устройства
Переподключение	Автоматическое: до 5 попыток, задержка 1 с, максимум 5 с

????????????????????????????????????



|
| disconnect

??????????

При подключении клиент передаёт в `auth` паспорт устройства:

```
{ "name": "tank-01", "type": "Robot", "telem_can": true, "video_can": true }
```

???????? ?? ????????? ? ???????

`heartbeat` (? ??????????????????)

Данные — список запросов. Робот возвращает словарь ответов (Socket.IO ack).

Запрос	Ответ	Побочный эффект
<code>"d_meta"</code>	Состояние: <code>{"sid": ..., "ready": "init" "ok", "telem_run": bool, "video_run": bool}</code>	—
<code>"Ready"</code>	<code>"ok"</code>	Робот переходит в состояние <code>ready = ok</code> и запускает потоки отправки телеметрии и видео
другое	—	Предупреждение в журнале

Пример ответа: `{"d_meta": {"sid": null, "ready": "ok", "telem_run": true, "video_run": true}, "Ready": "ok"}`

`token_ask` (? ??????????????????)

Запрос идентификатора. Робот возвращает значение `sid` из своего состояния.

`command`

```
{ "author": "<id автора>", "payload": { "<тип>": <команда> } }
```

Обрабатывается только в состоянии `ready = ok`. Типы `file` и `web_settings` обрабатываются SDK; остальные передаются в `@robot.com_handler`. См. [Обработка команд](#).

???????? ?? ?????? ? ????????

telem

```
{ "payload": { "battery": 11.7, "speed": 0.42, "imu": [0.01, -0.02, 9.81] } }
```

Полный текущий словарь телеметрии. Отправляется, только если с прошлой отправки что-то записывалось в телеметрию.

video

```
{ "payload": { "rgb": "<bytes: JPEG>", "rear": "<bytes: JPEG>" } }
```

Словарь «камера → кадр». Кадры — JPEG (качество 80) в виде бинарных данных Socket.IO. Для ключей, содержащих `led` или `rgb_strip`, значение передаётся как есть (без JPEG).

file

Ответ на команду `{"file": "log"}` или `{"file": "error"}`:

```
{ "payload": { "type": "log", "content": "<текст журнала>" } }
```

Команда `{"file": "clear"}` очищает журналы и ответа не отправляет.

???????

- Телеметрия и видео отправляются отдельными потоками с паузой `max_send_freq` секунд (по умолчанию 0,05 с).
- Отправляется только изменённое с момента прошлой отправки, поэтому фактическая частота ограничена частотой основного цикла робота.

???????????????? ?????????? ????????

Для отладки робота без платформы можно запустить имитатор сервера (Python, `pip install "python-socketio aiohttp"`):

```

import asyncio, socketio
from aiohttp import web

sio = socketio.AsyncServer(async_mode="aiohttp")
app = web.Application(); sio.attach(app)

@sio.on("connect", namespace="/robot")
async def connect(sid, environ, auth):
    print("Подключён:", auth)
    async def handshake():
        await asyncio.sleep(0.5)
        ans = await sio.call("heartbeat", ["d_meta", "Ready"], to=sid, namespace="/robot")
        print("heartbeat:", ans)
        await sio.emit("command", {"author": "tester", "payload": {"com": "forward"}},
                        to=sid, namespace="/robot")
    asyncio.create_task(handshake())

@sio.on("telem", namespace="/robot")
async def telem(sid, data):
    print("telem:", data["payload"])

@sio.on("video", namespace="/robot")
async def video(sid, data):
    print("video:", {cam: len(frame) for cam, frame in data["payload"].items()}, "байт")

web.run_app(app, port=8765)

```

Запустите его и укажите в `params.json` работа `"server": "http://localhost", "port": 8765`.

В целевой архитектуре платформы предусмотрены также передача видео по RTSP с преобразованием в WebRTC и подключение IoT-датчиков по MQTT. Описание — в книге «Сервер управления роботами», страница [Протоколы и потоки данных](#).

Revision #3

Created 2026-09-23 09:32:17 UTC by Маслов Алексей

Updated 2026-09-23 09:35:05 UTC by Маслов Алексей