liblog.sh — единая библиотека логирования для Nice OS Container

liblog.sh — Встроенная система логирования в НАЙС.ОС Container

liblog.sh — унифицированная, предсказуемая и минималистичная библиотека логирования для bash-скриптов и приложений в официальных образах **НАЙС.ОС Container**. Библиотека проста для новичков и достаточно гибка для опытных разработчиков и команд DevOps: стандартизирует логи, упрощает отладку, мониторинг и анализ инцидентов, не требует лишних зависимостей и работает только на русском языке без интернационализации.

1. Что это, где лежит и почему стоит использовать

liblog.sh — стандартная библиотека оболочки Bash, задающая единые правила логирования для всех контейнеров на базе НАЙС.ОС. Она **предустановлена** во всех базовых образах и доступна по фиксированному пути:

```
/nicesoft/niceos/scripts/liblog.sh
```

Ключевые особенности

- **Минимальные зависимости:** только штатные утилиты (`bash`, `date`, `mkdir`, `chmod`, `dirname`, `grep`, `sed`). Всё работает «из коробки».
- **Уровни:** `ERROR`, `WARN`, `INFO`, `DEBUG`, `TRACE` — гибкий контроль детализации.
- **Форматы вывода:** `plain` (человекочитаемый), `json` (машиночитаемый) и `both` (оба одновременно).
- **Назначения (sinks):** `stderr`, `file` или `both`. При записи в файл каталоги создаются автоматически, права устанавливаются согласно политике, при ошибке выполняется автоматический *fallback* на `stderr` с предупреждением.

- **Цвета:** «сырые» ANSI для консоли; уважаются `NO_COLOR` и `NICEOS_FORCE_COLOR`. В файлы ANSI-коды не попадают.
- **Дополнительно:** вывод PID и MODULE, склейка повторяющихся сообщений (rate-limit), поддержка многострочных сообщений, printf-форматтеры `*f()`, утилита `indent`.
- **Совместимость:** идеально вписывается в Docker/Kubernetes и другие оркестраторы: логи в `stderr` читаются через `docker logs` и `kubectl logs`.

Важно: не копируйте файл вручную — библиотека уже входит в базовые образы НАЙС.ОС Container. Для кастомных образов наследуйтесь от `niceos/container:latest` или совместимого.

В проектах с множеством скриптов и сервисов разрозненное логирование ведёт к хаосу. **liblog.sh** решает это: подключите её — и все компоненты начнут писать логи единообразно. Новичкам достаточно простого *import* и пары вызовов; профессионалы получают тонкую настройку через переменные окружения для интеграции с ELK/Graylog/Vector и др.

2. Базовая идея и принципы

1. **Единство.** Один механизм логирования для скриптов, entrypoint и сервисов упрощённая поддержка и быстрое понимание происходящего.
2. **Минимализм.** Никаких лишних зависимостей — меньше размер образов, выше скорость запуска, меньше рисков.
3. **Предсказуемость.** Одинаковое поведение в dev/stage/prod: форматы, уровни и назначения не меняются без явной конфигурации.
4. **Надёжность.** При ошибках записи в файл — предупреждение и автоматический переход на `stderr`.
5. **Чистота.** Цвета только в консоли; JSON строго экранирован (спецсимволы , \" и т.д.) и безопасен для парсеров.

Практика: в dev включайте `NICEOS_TRACE=true`, в prod — `NICEOS_QUIET=true`; следите за предупреждениями о *fallback* — это индикатор проблем с правами/томами.

3. Быстрый старт: подключение и первые логи

```
#!/usr/bin/env bash
set -Eeuo pipefail  # строгий режим
. /nicesoft/niceos/scripts/liblog.sh  # подключаем библиотеку
```

```
export NICEOS_MODULE="demo"  # имя модуля для контекста

info  "Приложение запускается"
warn  "Каталог /data отсутствует — создаю"
error "Не удалось подключиться к БД"

# printf-стиль форматирования
ELAPSED=1.234
infof  "Старт занял %0.3fs" "$ELAPSED"
errorf "Код ошибки: %d" 42

success "Инициализация успешна"
notice  "Важное уведомление"

# Диагностика текущей политики
niceos_log_self_check
```

Для многострочных сообщений просто передавайте строку с — библиотека корректно выведет каждую строку под общим заголовком. В интерактивном терминале можно включить цвета: `NICEOS_FORCE_COLOR=true`.

4. Как это устроено в стандартных образах НАЙС.ОС

- Все внутренние скрипты/утилиты (entrypoint, healthchecks) уже используют `liblog.sh`.
- Дефолт: формат `plain` и назначение `stderr` — идеально для `docker logs`/`kubectl logs`.
- По необходимости переключайтесь на `json` или `both` для интеграции с агрегаторами логов.
- В production допускается `both` (stderr+file) при наличии персистентного тома под журналы.

5. Настройка через переменные окружения

Переменная	Значения	По умолчанию	Описание / советы
`NICEOS_LOG_LEVEL`	`ERROR|WARN|INFO|DEBUG|TRACE`	`INFO`	Базовый уровень. Dev: `DEBUG`, Prod: `WARN` или `INFO`.
`NICEOS_LOG_FORMAT`	`plain|json|both`	`plain`	Выбор формата. `json` для парсеров; `both` — удобен при миграции.

Переменная	Значения	По умолчанию	Описание / советы
NICEOS_LOG_DEST	stderr|file|both	stderr	Куда писать. В оркестраторах обычно достаточно stderr.
NICEOS_LOG_FILE	путь	/var/log/niceos/containers.log	Файл-журнал для file/both. Убедитесь, что каталог доступен.
NICEOS_LOG_FILE_MODE	например 0640	0640	Права на файл при создании (безопасные настройки по умолчанию).
NICEOS_LOG_COLOR	auto|true|false	auto	Цвета в консоли. В файлах цвета никогда не пишутся.
NICEOS_FORCE_COLOR	true|false	false	Принудительное включение цветов (например, в интерактивных CI-TTY).
NICEOS_LOG_SHOW_PID	true|false	false	Добавляет pid=... в заголовок сообщения.
NICEOS_LOG_SHOW_MODULE	true|false	true	Отображать имя модуля (NICEOS_MODULE).
NICEOS_QUIET	true|false	false	Подавляет INFO/DEBUG/TRACE, оставляя WARN/ERROR.
NICEOS_DEBUG	true|false	false	Форсирует минимум DEBUG.
NICEOS_TRACE	true|false	false	Форсирует TRACE (максимальная детализация).
NICEOS_RATELIMIT_SAME_MS	целое 0	500	Интервал для склейки повторов одинаковых сообщений. 0 — отключить.

Переменная	Значения	По умолчанию	Описание / советы
`NICEOS_MODULE` / `MODULE`	строка	—	Имя модуля. Используйте уникальные значения (`web`, `api`, `worker`).

6. Поведение уровней, фильтрация и «тихий» режим

Приоритеты: `ERROR=0`, `WARN=1`, `INFO=2`, `DEBUG=3`, `TRACE=4`. Эффективный уровень увеличивается флагами `NICEOS_DEBUG`/`NICEOS_TRACE`, а `NICEOS_QUIET` скрывает «болтливые» уровни.

```
export NICEOS_QUIET=true
export NICEOS_LOG_LEVEL=INFO  # Результат: видны только WARN/ERROR
info "Это не выведется"
warn "Это выведется"
```

Совет: для циклов и сложных сценариев используйте `TRACE` только в dev; в prod оставляйте `INFO` или `WARN`.

7. Цвета: когда есть и когда нет

- Цвет применяется только к `plain`-сообщениям в `stderr` и только при разрешении политикой/TTY.
- `NO_COLOR` отключает цвет, `NICEOS_FORCE_COLOR=true` включает его независимо от TTY.
- В файл лог пишется всегда без ANSI — это гарантирует чистый парсинг.

```
export NICEOS_FORCE_COLOR=true
export NICEOS_LOG_COLOR=auto
info "Цветной вывод в TTY"
```

8. Назначения и fallback

- **stderr** — рекомендуемый дефолт для контейнеров.
- **file** — запись в `NICEOS_LOG_FILE`; каталог создаётся, права применяются.
- **both** — одновременно в консоль и файл (в файл — без цветов).

При недоступности файла библиотека сообщает и переключается на `stderr`:

```
⛔ Нет доступа к файлу логов: /var/log/niceos/containers.log — переключаемся на stderr
```

9. Форматы сообщений

9.1. Plain

Структура: `[MODULE] TS LEVEL pid=PID  MESSAGE`. Многострочные сообщения печатаются построчно с единым префиксом.

```
[web] 2025-09-20T12:34:56.789+0200 INFO pid=123  Сервер запущен на :8080
```

9.2. JSON

Машиночитаемый формат без ANSI-кодов, со строгим экранированием спецсимволов.

```
{"ts":"2025-09-20T12:34:56.789+0200","level":"INFO","module":"web","pid":123,"msg":"Сервер запущен на :8080"}
```

9.3. Both

Одновременный вывод plain+JSON в выбранные назначения.

10. Склейка повторов (rate-limit)

Повторяющиеся сообщения в течение интервала `NICEOS_RATELIMIT_SAME_MS` подавляются, а при смене сообщения/интервала выводится агрегатная запись:

```
🔁 Предыдущее сообщение повторилось 17 раз
```

Рекомендуемые значения: 200–1000 мс. `0` — полностью отключить (например, для тестов).

11. Примеры для Dockerfile: от простого к продвинутому

11.1. База (plain + stderr)

```
FROM niceos/container:latest
ENV NICEOS_MODULE=app
COPY entrypoint.sh /usr/local/bin/
ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]
```

Совет: добавьте HEALTHCHECK, используя `liblog.sh` в сценариях проверки.

11.2. JSON для агрегаторов

```
FROM niceos/container:latest
ENV NICEOS_LOG_FORMAT=json \
    NICEOS_LOG_DEST=stderr \
    NICEOS_MODULE=worker
```

11.3. Одновременно в консоль и в файл

```
FROM niceos/container:latest
RUN mkdir -p /var/log/niceos
ENV NICEOS_LOG_FORMAT=both \
    NICEOS_LOG_DEST=both \
    NICEOS_LOG_FILE=/var/log/niceos/containers.log \
    NICEOS_LOG_FILE_MODE=0640 \
    NICEOS_LOG_SHOW_PID=true \
    NICEOS_MODULE=service
```

11.4. Тихий образ (только WARN/ERROR)

```
FROM niceos/container:latest
ENV NICEOS_QUIET=true NICEOS_LOG_LEVEL=INFO
```

11.5. Включение TRACE на этапе сборки

```
FROM niceos/container:latest
ARG BUILD_TRACE=false
RUN if [ "$BUILD_TRACE" = "true" ]; then \
    export NICEOS_TRACE=true; \
    . /nicesoft/niceos/scripts/liblog.sh; \
    trace "Подробные логи сборки включены"; \
  fi
```

11.6. Многостадийная сборка с проверкой

```
# syntax=docker/dockerfile:1
FROM niceos/container:latest AS builder
ARG BUILD_DEBUG=false
RUN if [ "$BUILD_DEBUG" = "true" ]; then \
    export NICEOS_DEBUG=true NICEOS_MODULE=build; \
    . /nicesoft/niceos/scripts/liblog.sh; \
    debug "Сборка в DEBUG"; \
  fi

FROM niceos/container:latest AS runtime
ENV NICEOS_LOG_FORMAT=plain \
  NICEOS_LOG_DEST=stderr \
  NICEOS_MODULE=runtime
COPY --from=builder /app /app
ENTRYPOINT ["/app/run.sh"]
```

12. Пример entrypoint.sh

```
#!/usr/bin/env bash
set -Eeuo pipefail
. /nicesoft/niceos/scripts/liblog.sh

export NICEOS_MODULE="entrypoint"

niceos_log_self_check  # Диагностика

info "Старт приложения"
if ! command -v myapp >/dev/null; then
  error "myapp не найден в PATH"
  exit 127
fi

warn    "Используется конфигурация по умолчанию"
```

```
success "Инициализация завершена"
notice  "Переходим к основному процессу"
exec myapp "$@"
```

Подсказка: добавьте `trap 'error "Неперехваченная ошибка"' ERR` и используйте `indent` для красивого вывода подкоманд.

13. Таймстемпы и точность

- Формат времени: ISO-8601 с миллисекундами — `YYYY-MM-DDThh:mm:ss.SSS±zzzz`.
- Миллисекунды берутся через `date +%s%3N` с отказоустойчивыми *fallback*-механизмами (`/proc/uptime` и др.).
- Многострочные сообщения печатаются построчно с единым заголовком.

14. Безопасность, права и эксплуатация

- При создании файла-журнала применяются права `NICEOS_LOG_FILE_MODE` (по умолчанию `0640`).
- JSON-лог экранирует управляющие символы и кавычки, безопасен для парсеров.
- При невозможности писать в файл выполняется предупреждение и автоматический переход на `stderr`.
- Библиотека использует исключительно русский язык и игнорирует внешние переменные сторонних систем.

15. Производительность и надёжность

- Склейка повторов снижает «шум» и нагрузку на агрегаторы логов.
- Отсутствие внешних зависимостей ускоряет старт и повышает переносимость.
- Запись в файл — без цвета; формирование JSON — через лёгкие операции оболочки.

16. Практические рекомендации

- В Kubernetes/Docker начинайте с `stderr` или `both` + `json` для централизованного сбора.
- Всегда задавайте `NICEOS_MODULE` для быстрой фильтрации (например, `web`, `scheduler`, `worker`).
- Держите `NICEOS_RATELIMIT_SAME_MS` в диапазоне 200–1000 мс для сглаживания

бурстов.
- `DEBUG/TRACE` включайте на время диагностики; в проде — отключайте.
- Для файловых логов заранее создавайте каталог/том и настраивайте ротацию (host-side `logrotate` или сторонние механизмы).
- Не логируйте секреты/пароли; маскируйте конфиденциальные значения до передачи в `liblog.sh`.

17. Частые вопросы (FAQ)

Можно ли писать только JSON?

Да. Установите `NICEOS_LOG_FORMAT=json` и выберите нужное назначение (`stderr`/`file`/`both`).

Почему в файле нет цвета?

Это сознательная политика: ANSI-коды мешают анализу и парсингу. Цвет применяется только для интерактивной консоли.

Как отключить склейку повторов?

Установите `NICEOS_RATELIMIT_SAME_MS=0`.

Как указать модуль?

Задайте `NICEOS_MODULE` (или `MODULE`). Пример: `export NICEOS_MODULE=web`.

Что, если библиотека не найдена?

Убедитесь, что используете официальный базовый образ НАЙС.ОС Container и корректный путь `/nicesoft/niceos/scripts/liblog.sh`.

liblog.sh — ключ к стабильному и единообразному логированию в **НАЙС.ОС Container** без лишнего.