Руководство пользователя NORA

Версия: 2.0 **Дата:** 2026-05-20 **Правообладатель:** ООО «ТАИАРС»

1. Общие сведения

NORA — реестр артефактов для команд разработки. Программа обеспечивает хранение и кэширование библиотек, Docker-образов и иных программных компонентов, используемых при сборке приложений.

Данное руководство предназначено для разработчиков, которые используют NORA в качестве источника зависимостей.

2. Настройка рабочего окружения

2.1. npm / Node.js

Укажите NORA в качестве реестра:

```
npm config set registry http://nora.example.com:4000/npm
```

Или создайте файл `.npmrc` в корне проекта:

```
registry=http://nora.example.com:4000/npm
```

После этого все команды `npm install` будут загружать пакеты через NORA. При первом обращении NORA загрузит пакет из внешнего реестра (npmjs.org) и сохранит его в кэш. Последующие обращения обслуживаются из кэша.

2.2. Docker

```
docker login nora.example.com:4000
docker pull nora.example.com:4000/library/nginx:latest
docker push nora.example.com:4000/myteam/myapp:1.0.0
```

2.3. Maven

Добавьте репозиторий в `settings.xml`:

```
<mirrors>
  <mirror>
    <id>nora</id>
    <mirrorOf>central</mirrorOf>
    <url>http://nora.example.com:4000/maven2</url>
  </mirror>
</mirrors>
```

2.4. Python / pip

```
pip install --index-url http://nora.example.com:4000/simple flask
```

Или в `pip.conf`:

```
[global]
index-url = http://nora.example.com:4000/simple
```

2.5. Cargo / Rust

Настройка в `~/.cargo/config.toml`:

```
[registries.nora]
index = "sparse+http://nora.example.com:4000/cargo/"

[source.crates-io]
replace-with = "nora"
```

2.6. Go

Настройка через переменную окружения:

```
export GOPROXY=http://nora.example.com:4000/go,direct
```

Или в конфигурации проекта:

```
go env -w GOPROXY=http://nora.example.com:4000/go,direct
```

2.7. NuGet / .NET

Добавьте источник в `nuget.config`:

```
<configuration>
  <packageSources>
```

```
    <add key="nora" value="http://nora.example.com:4000/nuget/v3/index.json" />
  </packageSources>
</configuration>
```

Или через CLI:

```
dotnet nuget add source http://nora.example.com:4000/nuget/v3/index.json --name nora
dotnet restore
```

2.8. RubyGems / Bundler

```
gem sources --add http://nora.example.com:4000/gems/
```

Или в `Gemfile`:

```
source "http://nora.example.com:4000/gems/"
```

2.9. Terraform

Настройка в `~/.terraformrc`:

```
provider_installation {
  network_mirror {
    url = "http://nora.example.com:4000/terraform/"
  }
}
```

2.10. Ansible Galaxy

```
ansible-galaxy collection install community.general \
  --server http://nora.example.com:4000/ansible/
```

2.11. Pub / Dart / Flutter

```
export PUB_HOSTED_URL=http://nora.example.com:4000/pub
dart pub get
flutter pub get
```

2.12. Conan / C++

```
conan remote add nora http://nora.example.com:4000/conan/
conan install . --remote=nora
```

2.13. Helm

Helm использует OCI-протокол через Docker Registry API:

```
helm push mychart-0.1.0.tgz oci://nora.example.com:4000/helm
helm pull oci://nora.example.com:4000/helm/mychart --version 0.1.0
```

2.14. APT / Debian / Ubuntu

```
echo "deb http://nora.example.com:4000/apt stable main" | \
  sudo tee /etc/apt/sources.list.d/nora-proxy.list
sudo apt update
sudo apt install <package>
```

2.15. YUM / RPM / RHEL / CentOS

```
sudo yum-config-manager --add-repo http://nora.example.com:4000/yum/stable.repo
sudo yum install <package>
```

3. Публикация пакетов

3.1. npm

```
npm publish --registry http://nora.example.com:4000/npm
```

Требования: - Файл `package.json` с полями `name` и `version`. - Каждая версия публикуется однократно. Повторная публикация той же версии запрещена.

3.2. Docker

```
docker tag myapp:latest nora.example.com:4000/myteam/myapp:1.0.0
docker push nora.example.com:4000/myteam/myapp:1.0.0
```

3.3. Maven

```
mvn deploy
-DaltDeploymentRepository=nora::default::http://nora.example.com:4000/maven2
```

3.4. Raw (произвольные файлы)

```
# Загрузка
curl -X PUT --data-binary @release.tar.gz
```

```
http://nora.example.com:4000/raw/builds/release-1.0.tar.gz

# Скачивание
curl -O http://nora.example.com:4000/raw/builds/release-1.0.tar.gz
```

4. Работа в изолированной среде

Если сборочный сервер не имеет доступа к сети Интернет, используйте предварительное кэширование.

4.1. Кэширование зависимостей проекта

На машине с доступом к Интернету и NORA выполните:

```
nora mirror npm --lockfile package-lock.json --registry
http://nora.example.com:4000
```

После этого все зависимости из lockfile будут доступны через NORA, даже если связь с внешними реестрами отсутствует.

4.2. Кэширование всех версий пакета

```
nora mirror npm --packages lodash,express --all-versions --registry
http://nora.example.com:4000
```

Эта команда загрузит все опубликованные версии указанных пакетов.

5. Веб-интерфейс

NORA предоставляет веб-интерфейс для просмотра содержимого реестра:

```
http://nora.example.com:4000/ui/
```

Доступные функции: - Просмотр списка артефактов по протоколам. - Количество версий и размер каждого пакета. - Журнал последних операций. - Метрики загрузок. - Переключение языка интерфейса (русский / английский).

6. Документация API

Интерактивная документация API доступна по адресу:

```
http://nora.example.com:4000/api-docs
```

Формат: OpenAPI 3.0 (Swagger UI).

7. Аутентификация

Если администратор включил аутентификацию, для операций записи требуется токен.

7.1. Получение токена

```
curl -u admin:password http://nora.example.com:4000/auth/token
```

7.2. Использование токена

```
# npm
npm config set //nora.example.com:4000/npm/:_authToken TOKEN

# Docker
docker login nora.example.com:4000

# curl
curl -H "Authorization: Bearer TOKEN" http://nora.example.com:4000/npm/my-package
```

Операции чтения по умолчанию не требуют аутентификации (роль `read` назначается автоматически).

7.3. Единый вход (OIDC)

Если администратор настроил OIDC, аутентификация выполняется через провайдер единого входа (Keycloak, Google, Azure AD и др.). Токен передаётся в заголовке `Authorization: Bearer`.

8. Часто задаваемые вопросы

В: Что произойдёт, если внешний реестр (npmjs.org) станет недоступен? О: NORA продолжит обслуживать запросы из кэша. Пакеты, которые ранее не запрашивались, будут недоступны до восстановления связи. Для предотвращения такой ситуации используйте `nora mirror`. Если включён circuit breaker, NORA автоматически перейдёт в режим обслуживания

из кэша.

В: Можно ли публиковать приватные пакеты? О: Да. Пакеты, опубликованные через `npm publish` или `docker push`, сохраняются в локальном хранилище NORA и доступны всем пользователям данного экземпляра.

В: Как обновить кэш метаданных? О: Кэш метаданных npm обновляется автоматически по истечении TTL (по умолчанию 5 минут). Для немедленного обновления удалите файл `metadata.json` из каталога хранилища.

В: Поддерживаются ли scoped-пакеты npm (@scope/package)? О: Да, полностью. Например: `npm install @babel/core --registry http://nora.example.com:4000/npm`.

В: Сколько протоколов поддерживает NORA? О: 16 протоколов и форматов: Docker/OCI, Helm, npm, Maven, PyPI, Cargo, Go, NuGet, RubyGems, Terraform, Ansible, Pub (Dart/Flutter), Conan, APT (deb), YUM (rpm), а также Raw для произвольных файлов.