Amtiri

TECHNICAL INFORMATION

Configuration Reference

Generated on November 14, 2026

DataOrchester is configured in one file, `application.properties`, in its instance home. This reference lists every key it takes, with its default and the environment variable that overrides it; where the instance home is and what it holds; and the jar's commands.

The configuration file

`application.properties` is plain `key=value`, the same on every platform. The install package carries one that lists every key, grouped by concept, each with its default and a comment.

- **Before installing,** edit the package's copy: the install script copies it into the instance home.
- **Afterwards,** edit the home's copy and restart the service: the instance reads it at every start, and never reads the package's copy again. At an upgrade, `--config FILE` (`-Config` on Windows) replaces the home's copy on purpose.
- **A key left empty** takes the default its comment describes.
- **An environment variable,** where a key has one, overrides the file.
- **Secrets never go in it:** administrator passwords, activation codes and key store passwords reach the service by other means.

The instance reads its settings in this order, each overriding the one before:

1. `application.config`, the JSON file installations had before the install package. It is still read, so an older plant keeps its branding, but it is deprecated and the log says so at every start: move its keys, which have the same names, into `application.properties`.
2. `application.properties`.
3. The environment variables of the keys that have one.

Every key

Network

Key	Default	Environment	What it does
`server.port`	`8080`	`APP_PORT`	The HTTP port of the web interface
`server.address`	Empty: every interface	-	The interface to listen on; `127.0.0.1` when only a proxy on this machine connects
`application.trusted-proxies`	Empty: loopback only	`SITE_TRUSTED_PROXIES`	Reverse proxies allowed to report the client's address and protocol: IPv4 and IPv6 addresses, and IPv4 CIDR blocks, separated by commas. Loopback is always trusted
`application.public-base-url`	Empty: none	`SITE_PUBLIC_BASE_URL`	The address browsers use, such as `https://orchester.example.cl`. Needed for single sign-on; with `https`, session cookies travel over TLS only

HTTPS without a reverse proxy

The file carries this block commented out. Uncomment it in the home's copy to serve HTTPS on `server.port`:

```
server.ssl.enabled=true
server.ssl.key-store=tls/keystore.p12
server.ssl.key-store-type=PKCS12
server.ssl.key-store-password=${DATAORCHESTER_TLS_PASSWORD}
```

- `server.ssl.key-store` is a PKCS12 key store with the certificate and its private key. A relative path is resolved against the instance home, so `tls/keystore.p12` is in the home's `tls/` folder, where the service account must be able to read it.
- `server.ssl.key-store-password` takes the password from `DATAORCHESTER_TLS_PASSWORD`, in the service's environment: it is never written in the file. Set the variable before uncommenting the block - without the password, the web server cannot open the key store.
- Behind a reverse proxy, leave the block commented: the proxy terminates TLS.

The install scripts keep the service's environment when they run again, at an upgrade too: they rewrite only the activation code, and keep every variable they did not write. A value typed on a command line lands in the shell's history, so the commands below read the password first, without showing it.

Linux: a line in `/etc/data-orchester/data-orchester.env`, which only root can read (mode `0600`), then a restart:

```
read -rs TLS_PASSWORD
printf 'DATAORCHESTER_TLS_PASSWORD=%s\n' "$TLS_PASSWORD" | sudo tee -a /etc/data-orchester/data-orchester.env >
unset TLS_PASSWORD
sudo systemctl restart data-orchester
```

macOS: the `EnvironmentVariables` of the daemon's plist, which only root can read, then the daemon loaded again. The first `PlistBuddy` fails harmlessly when the plist already has them:

```
read -rs TLS_PASSWORD
sudo /usr/libexec/PlistBuddy -c "Add :EnvironmentVariables dict" /Library/LaunchDaemons/com.amtiri.dataorcheste
sudo /usr/libexec/PlistBuddy -c "Add :EnvironmentVariables:DATAORCHESTER_TLS_PASSWORD string $TLS_PASSWORD" /Li
unset TLS_PASSWORD
sudo launchctl bootout system/com.amtiri.dataorchester
sudo launchctl bootstrap system /Library/LaunchDaemons/com.amtiri.dataorchester.plist
```

Windows: the `Environment` value of the service's registry key, beside the entries already there. Then run the same package's `install.ps1 -Unattended` again - with `-Runtime FILE` on a machine with no network: it keeps the variable, closes the key to ordinary users and restarts the service. In an administrator PowerShell, from the extracted package:

```
$key = 'HKLM:\SYSTEM\CurrentControlSet\Services\DataOrchester'
$environment = @((Get-ItemProperty $key -Name Environment -ErrorAction SilentlyContinue).Environment | Where-Ob
$password = [Net.NetworkCredential]::new('', (Read-Host -AsSecureString 'Key store password')).Password
New-ItemProperty $key -Name Environment -PropertyType MultiString -Value ([string[]] ($environment + "DATAORCHE
Remove-Variable password
powershell -ExecutionPolicy Bypass -File .\install.ps1 -Unattended
```

Container: `-e DATAORCHESTER_TLS_PASSWORD`, by name, or under `environment:` from an `.env` file kept out of version control.

To change the password later, edit its line on Linux, use `Set` in place of the second `Add` on macOS, and run the same commands again on Windows.

Appearance

Key	Default	Environment	What it does
`application.title`	`Data Orchester`	-	The title of the sign-in card and the header
`application.subtitle`	Empty: none	-	The subtitle of the sign-in card and the header
`application.background-color`	Empty: the theme's	-	The page background, a CSS colour such as `#f4f4f0`
`application.bigLogo`	Empty: the DataOrchester logo	-	The logo on the sign-in card: a file in the home's `theme/` folder, SVG or bitmap, such as `theme/logo.svg`
`application.smallLogo`	Empty: the DataOrchester logo	-	The logo in the header, a file in `theme/` likewise
`application.theme`	Empty: none	-	An extra stylesheet in `theme/`, such as `theme/brand.css`, loaded after the built-in ones

> **Warning**
>
> Everything in `theme/` is served to browsers before anyone signs in: keep only theme files there.

Language and time

Key	Default	Environment	What it does
`application.locale`	`en`	-	The language of the interface for operators who have not chosen one: `en`, `es`, `fr`, `pt` or `de`. It also sets the engine's number format, so the formula `format()` writes decimal commas under `es`, `fr`, `pt` and `de`
`application.timezone`	Empty: the machine's zone	-	The plant's time zone, for module schedules, formulas and displays, such as `America/Santiago`
`server.timezone`	Empty: the machine's zone	-	The Java process's default time zone, which log timestamps use

Passwords

What a new local password must be. It is checked when a password is set or changed, never against one already stored, and only for local accounts: a directory or a single sign-on provider applies its own rules. See Sign-in and identity providers.

Key	Default	Environment	What it does
`application.password.min-length`	`12`	-	The fewest characters, a whole number from 1 to 128. Anything else is logged as an error, and 12 applies instead
`application.password.require-mixed-case`	`false`	-	`true`: a new password needs an uppercase and a lowercase letter
`application.password.require-digit`	`false`	-	`true`: a new password needs a digit
`application.password.require-special`	`false`	-	`true`: a new password needs a character that is neither a letter, a digit nor a space, such as `!`, `#` or `@`

NIST SP 800-63B-4 asks for at least 15 characters where a password is the only factor, as a local password is: set `application.password.min-length` to 15 to follow it. It forbids composition rules, because they push people towards predictable patterns such as `Password1!`, so turn the other three keys on only where your organization's policy demands them.

Licence

Key	Default	Environment	What it does
`orchester.portal.url`	`https://dataorchester.com`	`DATAORCHESTER_PORTAL_URL`	The Amtiri portal this instance takes its licence from
`orchester.site.url`	Empty: none	-	The address this instance reports to the portal, where Amtiri staff can open it
`orchester.licence.bindingPolicy`	Empty: `INSTALL_ID` in a container, `HARDWARE` otherwise	-	`HARDWARE` binds the licence to this machine; `INSTALL_ID` to this installation only, for virtual machines with no stable hardware identity

Engine and logging

Key	Default	Environment	What it does
`orchester.typing.strict`	`true`	-	Refuse a write whose value does not fit the variable's declared type; `false` warns and writes it
`orchester.transfer.maxBytes`	`268435456`	-	The largest configuration archive the Transfer pane accepts, in bytes: 256 MiB
`logging.level.root`	`INFO`	-	The level of `logs/server.log`: `ERROR`, `WARN`, `INFO` or `DEBUG`. `logs/security.log` records security events whatever this says

AI assistant

Key	Default	Environment	What it does
`application.assistant.enabled`	`true`	-	`false` hides the AI assistant on this instance, whatever its licence and the assistant's **Settings** say
`application.assistant.attachment-max-bytes`	`134217728`	-	The most bytes one conversation's attachments may hold together: 128 MiB. They are kept in memory only

Installation

`install.sh` and `install.ps1` read these keys; the instance ignores them.

Key	Default	Environment	What it does
`install.admin-account`	`ADMIN`	-	The first local administrator. The password is asked for or generated, and never stored here. `--admin` (`-Admin`) overrides it
`install.identity-provider`	Empty: a local administrator	-	A provider record file - a directory or single sign-on provider, with its administrators - installed instead of a local administrator; a relative path is resolved against the extracted package. See Sign-in and identity providers
`install.home-dir`	Empty: the platform's default	-	Where the instance home goes, such as a data drive for LOGGER history. Read at the first install only: an installed home never moves
`install.java`	Empty: a private Temurin 25	-	An existing Java 25 to run with, as `--java` (`-Java`)
`install.java-options`	Empty: Java's defaults	-	Java options for the service, such as `-Xmx2g` ; by default Java takes up to a quarter of the memory. Read from the home's copy whenever a script runs: after changing it, run the install script again

The instance home

The instance home is the instance's working directory: it reads and writes everything relative to it.

Platform	Instance home	Program: the jar and the Java runtime
Linux	`/var/lib/data-orchester`	`/opt/data-orchester`
macOS	`/Library/Application Support/DataOrchester`	`/Library/DataOrchester`
Windows	`C:\ProgramData\DataOrchester`	`C:\Program Files\DataOrchester`
Container	`/opt/orchester`	`/opt/orchester/app`

In the home	Contents
`application.properties`	The configuration file
`entities/`	Your configuration: peers, modules, variables, schemas - one file per entity
`logs/`	`server.log` , and `security.log` for security events
`loggers/`	Time-series data recorded by logger modules
`instance/`	The instance's own identity and licence state - readable by the service account alone, and not user-editable
`tmp/`	Java's temporary directory, where native libraries unpack
`theme/`	Your stylesheet and logos, named by the appearance keys; served to browsers before sign-in

An upgrade replaces the program and never touches the home; see Upgrading.

Entity files

Each entity (a peer, a module, a variable definition) is stored as one file under `entities/`, named by its type and id. You will not normally hand-edit these — the UI is the supported way to change configuration — but knowing the layout helps when scripting backups or diffing configuration between environments.

Where module-specific settings live

Each module type's settings, with the key each one is stored under in the module's entity file, are on its own page of the reference manual: the module catalogue lists every type and links to it. The settings every module has, its status and message variables among them, are in Settings every module shares.

A file a module needs travels inside its entity file rather than beside it: a CAN Signals module keeps the text of its DBC file there, so a backup, a transfer or a copy of `entities/` carries the database with it. The NMEA 2000 definitions the NMEA 2000 Bus module reads ship inside the engine, and nothing of them is kept in the home.

The jar's commands

The install scripts put the jar in the program directory as `data-orchester.jar`; the package holds it as `data-orchester-X.Y.Z.jar`.

Command	What it does
`java -jar data-orchester.jar`	Serves, from the current directory as its instance home
`java -jar data-orchester.jar identity-setup --local ACCOUNT [--must-change]`	Creates the first local administrator, run from the instance home. The password is read from `ORCHESTER_ADMIN_PASSWORD`, never from the command line; without it, the first sign-in defines one. With `--must-change`, they must replace it at their first sign-in
`java -jar data-orchester.jar identity-setup --provider FILE`	Installs a provider record instead: a directory or single sign-on provider, with its administrators
`java -jar data-orchester.jar version`	Prints the version
`java -jar data-orchester.jar selfcheck`	Proves the jar's libraries load on this machine; support may ask for its output

`identity-setup` acts only while no identity provider exists. It exits `0` when it installed, `1` when it refused, `2` on a usage error and `3` when there was nothing to install because an administrator already exists - the codes the install scripts share. On Windows the service runs the jar's `windows-service` command, which only the service manager starts.

Module status and messages

How a module reports its status and messages into variables of your own, and how to keep their history, is now described in Settings every module shares.

Next steps

- Upgrading
- Uninstalling
- Settings every module shares