**UnionVms**

**Config**

Marcus Hedenström

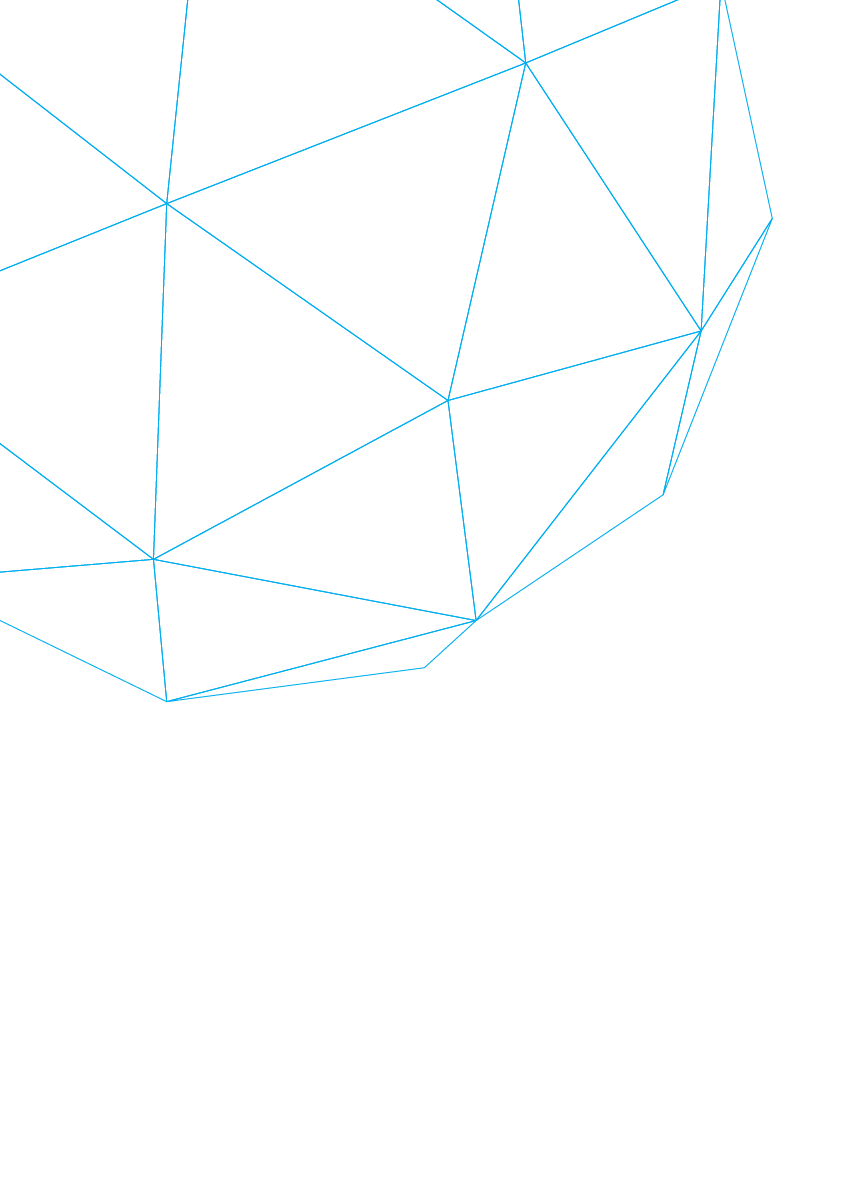


Table of contents

[Document history 3](#_Toc440640982)

[Module description/purpose 4](#_Toc440640983)

[Module dependencies 5](#_Toc440640984)

[JMS-Queue dependencies 5](#_Toc440640985)

[Datasources 5](#_Toc440640986)

[Tested compabilities 5](#_Toc440640987)

[Available database modules 6](#_Toc440640988)

[Available proxy modules 6](#_Toc440640989)

[Avaliable plugin modules 6](#_Toc440640990)

[Liquibase – database compability 6](#_Toc440640991)

|  |  |  |
| --- | --- | --- |
| Author | Changes | Date |
| Marcus Hedenström | Created document | 2016-01-15 |
|  |  |  |

# Document history

## Module description/purpose

The Config module is the main repository for all application settings. It stores both global and module-specific settings, and allows access to them by the application, typically through a REST interface. Settings can be accessed all at once (as a catalog), only the ones associated with a certain module including global settings or individually.

Individual module settings are synchronized with the Config module when either one is deployed to the application server. Newly deployed modules will initiate the synchronization process. Alternatively, if the Config is deployed last, it will notify all modules by posting on a topic, whereupon each module will initiate its own synchronization process accordingly. The process is as follows (using the Audit module as an example):

The Audit module sends a pull request to Config, requesting all of its settings.

If the Audit module is registered in the Config module, its current settings are returned. Otherwise, the Audit module is requested to, and subsequently will, upload its local settings to the Config module, which registers them for future use.

For the remaining runtime, the modules will access their settings locally. If settings are updated in the Config module, they will be pushed to the corresponding module on the aforementioned topic, and stored. It is also possible for modules to push settings to the Config module on the message queue.

Another feature of the Config module is keeping track of the availability of other modules, by collecting the latest timestamp of pings repeatedly transmitted by the modules on the Config module’s input queue. A summary of the state of each module can be retrieved through the Config modules REST interface by the application.

Much of this functionality (including synchronization, receiving changes during runtime and transmitting ping messages) is implemented in a separate project called uvms-config, which can be added as a dependency to modules. It requires the module to implement two interfaces for sending and receiving messages, as well as an interface that acts as a source for customizing the internal module name and a list of setting keys specific to that module. It also contains the classes used to access to the module’s local settings store, to reduce code duplication across modules.

## Module dependencies

|  |  |
| --- | --- |
| Name | Description |
| Audit | Logs database changes. |
| User | Authorizes REST requests according to user privileges. |

### JMS-Queue dependencies

The jndi name example is taken from wildfly8.2 application server

|  |  |  |
| --- | --- | --- |
| Name | Jndi name example | Description |
| UVMSConfigEvent | java:/jms/queue/UVMSConfigEvent | Config request listening queue |
| UVMSConfig | java:/jms/queue/UVMSConfig | Resposne queue |
| UVMSConfigModel | java:/jms/queue/UVMSConfigModel | Request queue of datasource |
| ConfigStatus | java:/jms/topic/ConfigStatus | Topic for posting to other modules |
| UVMSAuditEvent | java:/jms/queue/UVMSAuditEvent | Request queue for audit events |

### Datasources

The jndi name example is taken from wildfly8.2 application server

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Jndi name example |  |  |
| uvms\_config | Java:jboss/datasources/uvms\_config |  |  |

## Tested compabilities

|  |  |
| --- | --- |
| Applicationserver | * Wildfly 8.1 * Wildfly 8.2 |
| Database | * Postgres 9.4.4 |
| JMS provider | * ActiveMq 5.0 |

## Available database modules

All database modules can be found under the DB folder

|  |  |
| --- | --- |
| Name | Purpose |
| config-dbaccess | Storing settings, accessed by Config module |

### Available proxy modules

All proxy modules can be found under the PROXY folder

|  |  |
| --- | --- |
| Name | Purpose |
| N/A |  |

### Avaliable plugin modules

All plugin modules can be found under the PLUGIN folder

|  |  |
| --- | --- |
| Name | Purpose |
| N/A |  |

### Liquibase – database compability

All Liquibase scripts can be found under the LIQUIBASE folder

|  |  |
| --- | --- |
| Name | Version |
| Postgres | 9.4.4 |