Центр Управления Libercat
Руководство пользователя

ЦУ Libercat 1.0.0 | Версия документа 1.0 | 26.12.2024

AXIOM JDK

Программное обеспечение АКСИОМ содержит программное обеспечение с открытым исходным кодом. Дополнительная информация о коде сторонних разработчиков доступна на сайте https://axiomjdk.ru/third_party_licenses. Для дополнительной информации о том, как получить копию исходного кода, можно обратиться по адресу info@axiomjdk.ru.

ДАННАЯ ИНФОРМАЦИЯ МОЖЕТ ИЗМЕНЯТЬСЯ БЕЗ ПРЕДВАРИТЕЛЬНОГО УВЕДОМЛЕНИЯ. АКСИОМ ПРЕДОСТАВЛЯЕТ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ "КАК ЕСТЬ" БЕЗ КАКИХ-ЛИБО ГАРАНТИЙ, АКСИОМ ПРЯМО ОТКАЗЫВАЕТСЯ ОТ ВСЕХ ПОДРАЗУМЕВАЕМЫХ ГАРАНТИЙ, ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯСЬ ПОДРАЗУМЕВАЕМЫМИ ГАРАНТИЯМИ ТОВАРНОЙ ПРИГОДНОСТИ И ПРИГОДНОСТИ ДЛЯ ОПРЕДЕЛЕННОЙ ЦЕЛИ.

АКСИОМ НИ ПРИ КАКИХ ОБСТОЯТЕЛЬСТВАХ НЕ НЕСЕТ ОТВЕТСТВЕННОСТИ ЗА ЛЮБЫЕ КОСВЕННЫЕ, СЛУЧАЙНЫЕ, СПЕЦИАЛЬНЫЕ, ШТРАФНЫЕ ИЛИ КОСВЕННЫЕ УБЫТКИ, ИЛИ УБЫТКИ ОТ ПОТЕРИ ПРИБЫЛИ, ДОХОДА, ДАННЫХ ИЛИ ИСПОЛЬЗОВАНИЯ ДАННЫХ, ПОНЕСЕННЫЕ ВАМИ ИЛИ ЛЮБОЙ ТРЕТЬЕЙ СТОРОНОЙ, БУДЬ ТО В РЕЗУЛЬТАТЕ ДЕЙСТВИЯ ДОГОВОРА ИЛИ ДЕЛИКТА, ДАЖЕ ЕСЛИ АКСИОМ БЫЛО ПРЕДУПРЕЖДЕНО О ВОЗМОЖНОСТИ ТАКИХ УБЫТКОВ.

Использование любого программного продукта АКСИОМ регулируется соответствующим лицензионным соглашением, которое никоим образом не изменяется условиями данного уведомления. Программные продукты и фирменные наименования: Axiom JDK, Axiom JDK Pro, Axiom Runtime Container Pro, Axiom Linux, Libercat, Libercat Certified и АКСИОМ принадлежат АКСИОМ и их использование допускается только с разрешения правообладателя.

Товарный знак Linux® используется в соответствии с сублицензией от Linux Foundation, эксклюзивного лицензиата Линуса Торвальдса, владельца знака на всемирной основе. Java и OpenJDK являются товарными знаками или зарегистрированными товарными знаками компании Oracle и/или ее аффилированных лиц. Другие торговые марки являются собственностью их соответствующих владельцев и используются только в целях идентификации.

Содержание

1. Введение

Центр управления Libercat предназначен для наблюдения за серверами приложений, работающими на базе Libercat 9.x, Libercat 10.x, Libercat EE 8.x и Libercat EE 9.x и для управления ими.

На современных предприятиях одновременно может эксплуатироваться большое количество независимых серверов приложений различных версий с различным набором приложений, библиотек, источников данных и других необходимых сущностей. **Центр управления Libercat** позволяет серверам приложений регистрироваться и дает возможность администратору иметь единую картину о состоянии серверов, изменять это состояние в зависимости от изменяющихся требований.

2. Архитектура

Центр управления Libercat (ЦУ) - это независимое приложение, которое требует для работы наличие Java. Кроме того, в поставке центра управления есть **агент** для серверов приложений Libercat (EE), который позволяет ЦУ взаимодействовать с серверами приложений и предоставляет ему необходимые данные о серверах, а также выполняет управляющие команды от ЦУ.

Агент центра управления (агент ЦУ) - это отдельный `.jar` архив, собранный с помощью JDK 8 и доступный для исполнения в составе любого из поддерживаемых серверов, работающих на Java 8 и новее.

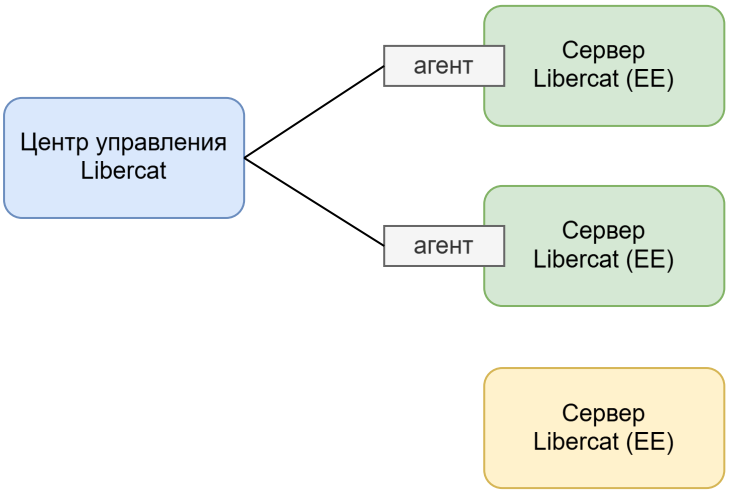


Рис 1. Взаимодействие центра управления Libercat и серверов приложений.

Рисунок 1 демонстрирует, что центр управления Libercat берет под свой контроль только те сервера приложений Libercat (EE), которые запущены с соответствующим агентом. За серверами приложений, в которых отсутствует агент ЦУ, не производится наблюдение и не осуществляется управление.

Более детальная архитектура ЦУ Libercat отражена на следующем рисунке:

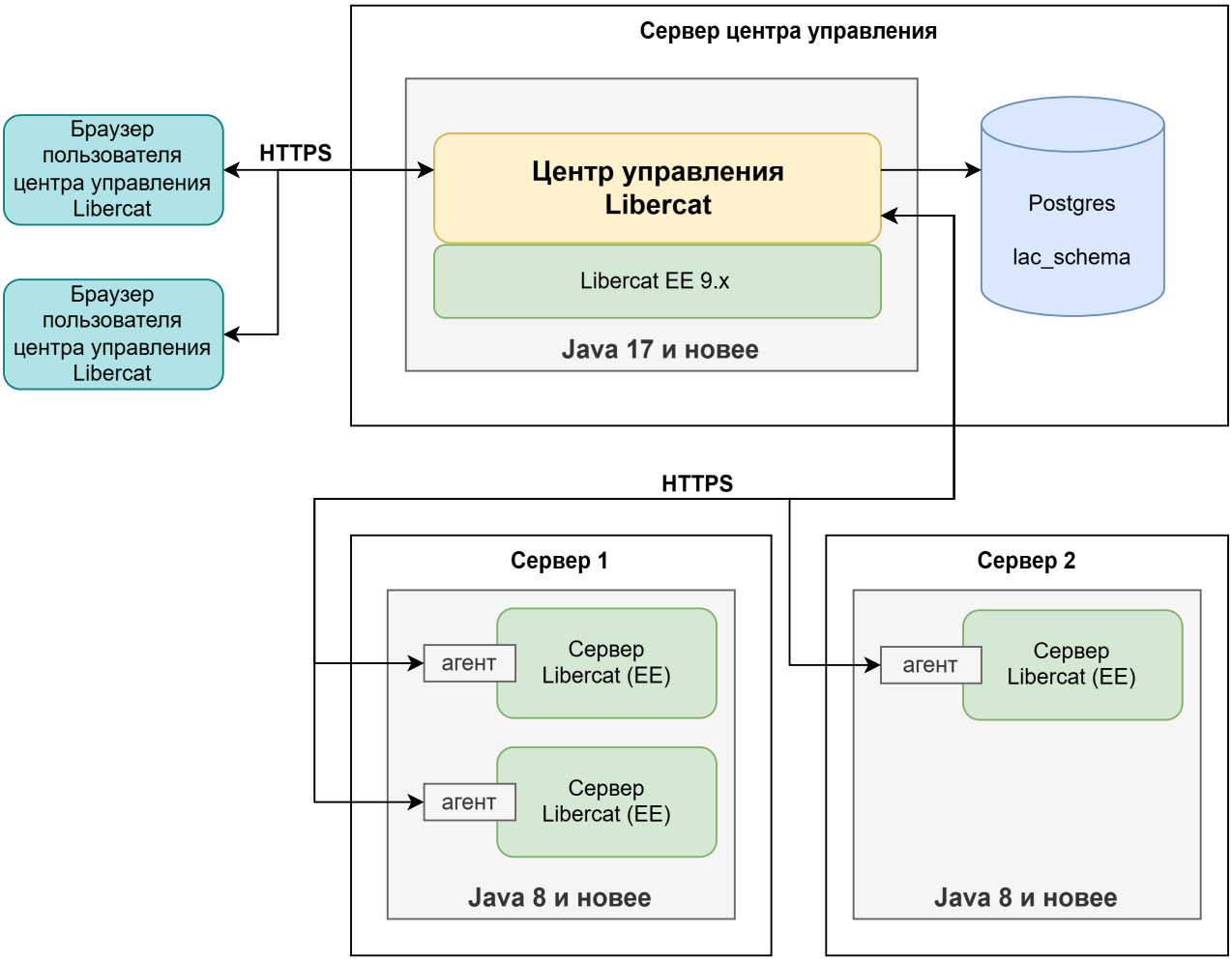


Рис 2. Архитектура центра управления Libercat.

В состав центра управления Libercat входит сервер приложений Libercat EE 9.x, в котором исполняется ЦУ. Для взаимодействия с пользователями и наблюдаемыми серверами приложений используется протокол HTTPS.

Центр управления Libercat использует базу данных PostgreSQL для своей консистентной работы. В базе хранятся данные о наблюдаемых серверах, а также системные настройки ЦУ.

3. Поставка

Центр управления Libercat поставляется как единый архив в формате **tar.gz** для Unix платформ и в формате **zip** для Windows системы. Оба архива содержат:

- Приложение Центр Управления Libercat;
- Агент ЦУ Libercat для управляемых серверов и его документация;
- Конфигуратор ЦУ Libercat;
- Libercat EE версии 9.1.3-2;
- Драйвер для работы с PostgreSQL версии 42.7.4.

4. Первичная конфигурация

Перед первичным запуском центра управления Libercat необходимо произвести его настройку при помощи встроенного конфигуратора.

Предварительные действия для запуска конфигуратора:

1. Распакуйте соответствующий архив (**tar.gz** или **zip**) в зависимости от платформы, где будет работать ЦУ, в папку **<lac_dir>**.
2. Установите LTS JRE версии не ниже 17 (см. Руководство по установке для соответсвующий версии).
3. Установите или воспользуйтесь уже существующей системой PostgreSQL. Создайте пустую базу данных в PostgreSQL (например, *lac*), к которой будет иметь доступ административный пользователь (например, *lac_db_admin*) с правами создания DDL объектов. Это пользователь будет использован единожды в процессе конфигурации системы и не будет нигде сохранен. Структура базы данных, конфигурационные данные, роли и пользователь для работы с центром управления (если в нем есть необходимость) будут созданы в процессе конфигурации.
4. Подготовьте файл с настройками ЦУ Libercat (см. Настройки центра управления Libercat).

После проведенных действий можно запустить конфигуратор (см. Запуск конфигуратора).

Настройки центра управления Libercat

Настройки центра управления Libercat находятся в конфигурационном файле, который поступает на вход конфигуратора. Этот файл может содержать параметры из разделов Основные конфигурационные параметры и Дополнительные конфигурационные параметры.

Пример конфигурационного файла можно найти в **<lac_dir>/conf/configuration.properties**. Можно отредактировать и использовать этот файл для передачи конфигурационных параметров (этот файл используется по умолчанию) или создать другой, указав к нему путь во время запуска конфигуратора.

Основные конфигурационные параметры

Существует ряд параметров, которые должны быть обязательно указаны в конфигурационном файле и без которых конфигуратор не сможет работать или будет работать неправильно.

Сервер

- `lac-init.server.host` - хост, на котором работает центр управления Libercat. Необязательный параметр, значение по умолчанию *localhost*.
- `lac-init.server.front-port` - основной HTTPS порт центра управления Libercat, на котором находится UI ЦУ. Необязательный параметр, значение по умолчанию *443*.
- `lac-init.server.agent-port` - HTTPS порт центра управления Libercat, используемый агентами управляемых серверов. Необязательный параметр, значение по умолчанию *8443*.
- `lac-init.server.shutdown-port` - порт для окончания работы центра управления Libercat. Необязательный параметр, значение по умолчанию *8005*.

База данных

- `lac-init.database.admin.username` - логин административного пользователя базы данных с правами создания DDL объектов.
- `lac-init.database.url` - URL к PostgreSQL базе данных, которая будет использоваться центром управления. Например, *jdbc:postgresql://localhost:5432/lac*.

Сертификат сервера

Центр управления Libercat использует HTTPS соединения для работы с пользователем и управляемыми серверами приложений. Для работы по HTTPS необходимо предоставить сертификат сервера. Конфигуратор позволяет использовать или существующий сертификат, находящийся в keystore, или сгенерировать self-signed сертификат по заданным параметрам.

`lac-init.certificate.generation` - будет ли сертификат сгенерирован конфигуратором или будет использоваться существующий. Возможные значения: *true, false*.

Сохранение готового сертификата

- `lac-init.certificate.keystore.file` - путь до существующего keystore, в котором находится сертификат сервера, если `lac-init.certificate.generation=false`. Если `lac-init.certificate.generation=true`, параметр игнорируется.
- `lac-init.certificate.keystore.type` - тип существующего keystore, в котором находится сертификат сервера, если `lac-init.certificate.generation=false`.

  Возможные значения: *JKS*. Если `lac-init.certificate.generation=true`, параметр игнорируется.

- `lac-init.certificate.alias` - имя сертификата в существующем keystore, в котором находится сертификат сервера, если `lac-init.certificate.generation=false`. Если `lac-init.certificate.generation=true`, параметр игнорируется.

Генерация сертификата

- `lac-init.certificate.generation.bits` - количество бит сгенерированного ключа, которым будет подписан сертификат. Необязательный параметр, значение по умолчанию берется из `lac.certificate.generation.bits` равное *2048*. Параметр будет использован, только если `lac-init.certificate.generation=true`, в другом случае игнорируется.
- `lac-init.certificate.generation.validity` - количетво дней, в течении которых сертификат будет достоверным начиная с сегодняшнего дня. Необязательный параметр, значение по умолчанию берется из `lac.certificate.generation.validity` равное *365*. Параметр будет использован, только если `lac-init.certificate.generation=true`, в другом случае игнорируется.
- `lac-init.certificate.generation.subject.*` - поля субъекта сертификата. На данный момент поддерживаются только следующие поля: *CN, OU, O, L, ST, C*. Остальные поля будут проигнорированы. Поле *CN* должно быть обязательно заполнено, остальные на усмотрение администратора. Параметр будет использован, только если `lac-init.certificate.generation=true`, в другом случае игнорируется.

Пример:

```
lac-init.certificate.generation.subject.CN = localhost
lac-init.certificate.generation.subject.OU = OrgUnit
lac-init.certificate.generation.subject.O = Organization
lac-init.certificate.generation.subject.L = St.Petersburg
lac-init.certificate.generation.subject.ST = St.Petersburg
lac-init.certificate.generation.subject.C = RU
```

Механизм аутентификации и авторизации

`lac.users.auth-type.allowed` - допустимые типы аутентификации и авторизации в ЦУ. Возможные значения: *SELF, LDAP* или комбинация этих значений через запятую. Необязательный параметр, значение по умолчанию *SELF*.

Аутентификация и авторизация в ЦУ может быть осуществлена, как через внутренний механизм (что соответствует значению *SELF*), так и через существующую в организации LDAP систему (что соответствует значению *LDAP*).

Администратор центра управления

Если механизм аутентификации и авторизации определенный в `lac.users.auth-type.allowed`, содержит *SELF* значение, то следующие поля могут быть заданы, в противном случае они будут проигнорированы.

При этом если `lac.users.auth-type.allowed` содержит только *SELF* значение, то следующие поля должны быть определены, т.к. это единственный способ задать первоначального пользователя системы. Такой пользователь создается с ролью администратора. Последующие пользователи могут быть добавлены в систему при помощи первоначального пользователя.

- `lac-init.users.admin.login` - электронная почта первоначального административного пользователя, который будет считаться его логином. Если предполагается использование электронной почты для смены паролей пользователей, то необходимо, чтобы это поле представляло собой реальный электронный адрес с возможностью получать письма.
- `lac-init.users.admin.name` - имя первоначального административного пользователя, которое будет отображаться в UI центра управления Libercat.

Соответствие LDAP групп

Если механизм аутентификации и авторизации определенный в `lac.users.auth-type.allowed`, содержит *LDAP* значение, то следующие поля должны быть заданы, в противном случае они будут проигнорированы.

Для использования аутентификации и авторизация через LDAP, необходимо задать соответствие между группами LDAP и ролями ЦУ.

`lac-init.users.ldap-config.lac-roles.<LDAP_GROUP_NAME>` - соответствие LDAP группы роли или списку ролей в ЦУ.

Параметр `lac-init.users.ldap-config.lac-roles.<LDAP_GROUP_NAME>` должен быть установлен для каждой LDAP группы, которая используется при авторизации пользователей в ЦУ. Этот параметр может содержать значения *ADMIN*, *VIEWER*, *MODERATOR* или любую комбинацию этих значений через запятую. Несколько LDAP групп могут соответствовать нескольким ролям в ЦУ.

> **Совет:**
>
> Включите ЦУ роль *VIEWER* в каждую LDAP группу, которая соответствует ЦУ роли *MODERATOR*. См. пример ниже.

Пример:

```
lac-init.users.ldap-config.lac-roles.ADMIN_GR = ADMIN
lac-init.users.ldap-config.lac-roles.VIEWER_GR = VIEWER
lac-init.users.ldap-config.lac-roles.MODERATOR_GR = VIEWER,MODERATOR
lac-init.users.ldap-config.lac-roles.ANOTHER_GR = ADMIN,VIEWER
```

Конфигурация LDAP

Если механизм аутентификации и авторизации, определенный в `lac.users.auth-type.allowed`, содержит *LDAP* значение, то поля из этого раздела тоже должны быть заданы в соответствии с настройками LDAP сервера.

Все конфигурационные параметры, описанные в этом разделе отражают соответствующие поля класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`, принадлежащего Jakarta EE Platform API.

- `lac.users.ldap-identity.url` - URL к LDAP серверу, который будет использоваться для аутентификации и авторизации в ЦУ. Обязательный параметр. Этот параметр имеет то же функциональное назначение, что и поле url класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.bind-dn` - заданное имя приложения или административного пользователя, которое будет использовать соединения с LDAP сервером, чтобы осуществить поиск. Этот параметр имеет то же функциональное назначение, что и поле `bindDn` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.bind-dn-password` - пароль приложения или административного пользователя LDAP, заданного в `lac.users.ldap-identity.bind-dn`. Этот параметр имеет то же функциональное назначение, что и поле `bindDnPassword` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.caller-search-base` - поисковая база, для нахождения пользователей, которые аутентифицируются и авторизируются в ЦУ. Этот параметр имеет то же функциональное назначение, что и поле `callerSearchBase` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.caller-search-filter` - фильтр поиска для нахождения пользователей. Этот параметр имеет то же функциональное назначение, что и поле `callerSearchFilter` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.caller-search-scope` - область поиска пользователя. Этот параметр имеет то же функциональное назначение, что и поле `callerSearchScopeExpression`

класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.

- `lac.users.ldap-identity.group-search-base` - поисковая база, для нахождения групп, в которых состоят аутентифицирующиеся и авторизирующиеся в ЦУ пользователи. Этот параметр имеет то же функциональное назначение, что и поле `groupSearchBase` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.group-search-filter` - фильтр поиска для нахождения групп. Этот параметр имеет то же функциональное назначение, что и поле `callerGroupFilter` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.group-search-scope` - область поиска групп. Этот параметр имеет то же функциональное назначение, что и поле `groupSearchScopeExpression` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.group-name-attribute` - имя атрибута объекта группы в LDAP, который представляет собой название имени группы. Необязательный параметр, значение по умолчанию `cn`. Этот параметр имеет то же функциональное назначение, что и поле `groupNameAttribute` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.group-member-attribute` - имя атрибута объекта группы в LDAP, который представляет собой членов группы. Необязательный параметр, значение по умолчанию `member`. Этот параметр имеет то же функциональное назначение, что и поле `groupMemberAttribute` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.group-member-of-attribute` - имя атрибута объекта человека в LDAP, который представляет собой список групп, к которым человек принадлежит. Необязательный параметр, значение по умолчанию `memberOf`. Этот параметр имеет то же функциональное назначение, что и поле `groupMemberOfAttribute` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.read-timeout` - время ожидания ответа LDAP сервера, в миллисекундах. Необязательный параметр, значение по умолчанию 0 (ждать вечно). Этот параметр имеет то же функциональное назначение, что и поле `readTimeoutExpression` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.
- `lac.users.ldap-identity.max-results` - максимально количество объектов, возвращаемое сервером LDAP. Необязательный параметр, значение по умолчанию 1000. Этот параметр имеет то же функциональное назначение, что и поле `maxResultsExpression` класса `jakarta.security.enterprise.identitystore.LdapIdentityStoreDefinition`.

Пример:

```
lac.users.ldap-identity.url = ldaps://localhost:636
```

```
lac.users.ldap-identity.bind-dn = cn=bind,cn=Users,dc=localhost
lac.users.ldap-identity.bind-dn-password = password
lac.users.ldap-identity.caller-search-base = ou=Lac Users,dc=localhost
lac.users.ldap-identity.caller-search-filter =
(&(sAMAccountName=%s)(objectClass=user))
```

Дополнительные конфигурационные параметры

Также существует ряд параметров для настройки центра управления, которые либо не обязательны (в зависимости от использованной конфигурации), либо имеют значения по умолчанию.

Шифрование параметров

ЦУ Libercat может хранить чувствительную информацию в базе данных, например, пароли. Чтобы чувствительная информация не находилась в базе данных в открытом виде, можно воспользоваться шифрованием.

Имейте в виду, что информация из конфигурационного файла сохраняется в базе данных в том виде, в котором она представлена. Конфигуратор самостоятельно ничего не шифрует. Если в конфигурационном файле есть чувствительная информация, она должна быть помещена туда в зашифрованном виде.

Зашифрованная информация выглядит следующим образом:

`ENC(кодировка *base64* от зашифрованной информации)`

Пример:

`lac.users.notifications.mail.password = ENC(mutzB42pn+ynhE/uSC6nGA==)`

- `lac.secure.master-key` - абсолютный путь до защищенного файла, содержащего секретный ключ шифрования. Файл должен быть доступен для чтения пользователю операционной системы, под которым запущен ЦУ Libercat.
- `lac.secure.algorithm` - алгоритм шифрования. На данный момент поддерживаются только PBE алгоритмы, существующие в используемой JRE. Необязательный параметр, значение по умолчанию *PBEWithMD5AndDES*.
- `lac.secure.pbe.salt` - соль для PBE алгоритмов, используемая во время шифрования. Необязательный параметр, значение по умолчанию *FyGfcmYO*.
- `lac.secure.pbe.iterations` - количество итераций PBE алгоритма при шифровании. Необязательный параметр, значение по умолчанию *1000*.

Управление пользователями

Настройки почты

Для того, чтобы пользователи ЦУ могли самостоятельно задать себе первоначальный пароль и поменять его в случае утери, необходимо настроить отправку автоматических уведомлений через электронную почту. Если настройки почты не были заданы, то эти операции можно будет осуществить только через пользователя ЦУ, который обладает ролью **администратора**.

- `lac.users.notifications.mail.username` - почтовый ящик для отправки уведомлений ЦУ Libercat
- `lac.users.notifications.mail.password` - пароль от почтового ящика, предназначенного для отправки уведомлений ЦУ Libercat
- `lac.users.notifications.setup.*` - набор параметров, соответствующих Jakarta Mail 2.0 спецификации для отсылки уведомлений ЦУ. Имя каждого параметра строится по принципу `префикс.полное-имя-параметра-по-jakarta-mail`, где префикс - это `lac.users.notifications.setup`.

Пример:

```
lac.users.notifications.mail.username = admin@server.ru
lac.users.notifications.mail.password = password
lac.users.notifications.setup.mail.smtp.host = smtp.server.ru
lac.users.notifications.setup.mail.smtp.port = 465
lac.users.notifications.setup.mail.smtp.auth = true
lac.users.notifications.setup.mail.smtp.ssl.enable = true
```

Управление аутентификацией и сессиями

- `lac.users.session.timeout` - время *в секундах*, в течение которого сеанс пользователя остается активным после входа в ЦУ. 0 или отрицательное значение означает, что сеанс никогда не должен прерываться. Необязательный параметр, значение по умолчанию *10800 (3 часа)*.
- `lac.users.login.failure-count` - максимальное количество неудачных попыток входа, после которого пользователь будет заблокирован на ac.users.login.lock-out-time время. Необязательный параметр, значение по умолчанию *3*.
- `lac.users.login.lock-out-time` - время блокировки пользователя *в секундах* после lac.users.login.failure-count неудачных попыток входа. Необязательный параметр, значение по умолчанию *3600 (1 час)*.

Правила составления и хранения паролей

- `lac.users.password.length` - минимальная длина пароля пользователя ЦУ. Необязательный

параметр, значение по умолчанию *8*.

- `lac.users.password.alphabet` - регулярное выражение, которое определяет допустимый алфавит используемый в пароле пользователя ЦУ. Необязательный параметр, значение по умолчанию [а-яА-Яа-zA-Z0-9~!@#$%^&(){}?',=+№;:_|<>\\"\\[\\]*\\-\\.\\/\\\\]*. Если установить это значение пустым, то проверка на алфавит не будет осуществляться.
- `lac.users.password.rules` - группы символов, один или более элементов которых должны обязательно присутствовать в пароле. Необязательный параметр, значение по умолчанию [a-z],[A-Z],[0-9],[@#$%^&*].
- `lac.users.password.expire` - срок действия пароля *в днях*, после которого пользователю необходимо его заменить. Необязательный параметр, значение по умолчанию *45*.
- `lac.users.password.old-values` - количество последних старых паролей пользователя, значения которых запрещены к использованию. Необязательный параметр, значение по умолчанию *12*.

Сброс паролей

lac.users.password.reset-token.expire - время *в минутах*, в течении которого токен сброса пароля считается действительным. Необязательный параметр, значение по умолчанию *60 (1 час)*.

Наблюдение за серверами

Сертификаты

Как ЦУ Libercat использует HTTPS соединение для работы с управляемыми серверами приложений, так и агент управляемых серверов использует HTTPS соединения для доступа к данным серверов. Для работы по HTTPS необходимо предоставить сертификат управляемого сервера.

При подключении управляемого сервера можно предоставить существующий сертификат, находящийся в keystore, или сгенерировать self-signed сертификат по заданным параметрам.

- `lac.certificate.generation.bits` - количество бит сгенерированного ключа, которым будет подписан сертификат. Необязательный параметр, значение по умолчанию *2048*.
- `lac.certificate.generation.validity` - количетво дней, в течении которых сертификат будет достоверным начиная с сегодняшнего дня. Необязательный параметр, значение по умолчанию *365*.

Частота обновления информации

`lac.servers.data.update-timeout` - время ограничения запроса данных с управляемых серверов *в секундах*. Во избежание большой нагрузки на управляемые сервера, ЦУ сокращает число непосредственных запросов к серверам путем кэширования информации о них. Необязательный

параметр, значение по умолчанию *300 (5 минут)*.

Списки наблюдаемых переменных

Центр управления Libercat представляет информацию о системных свойствах и переменных окружения на управляемых серверах. Если пользователям необходима эта информация, то надо задать соответствующие списки переменных через запятую.

- `lac.servers.data.system-properties` - список наблюдаемых системных свойств.
- `lac.servers.data.environment-variable` - список переменных окружения.

Запуск конфигуратора

Конфигуратор центра управления Libercat находится в папке *<lac_dir>/lib* с именем *lac-configurator-<VERSION>.jar* и запускается как обычное Java приложение командой *java*.

При запуске конфигуратора на вход должны быть поданы чувствительные конфигурационные параметры в виде аргументов конфигуратора.

Аргументы конфигуратора

Для работы конфигуратора необходимо задать ряд паролей, которые поступают непосредственно на вход конфигуратора и не сохраняются в конфигурационном файле.

Также на вход конфигуратора может быть подан путь к файлу с конфигурационными параметрами, если он находится в нестандартном месте.

- --database-password <arg> - пароль административного пользователя базы данных с правами создания DDL объектов.
- --admin-password <arg> - пароль первоначального административного пользователя ЦУ, если 'lac.users.auth-type.allowed' содержит SELF значение и 'lac-init.users.admin.login' не пустой, в противном случае аргумент игнорируется.
- --keystore-password <arg> - пароль от существующего keystore, заданного в 'lac-init.certificate.keystore.file', если 'lac-init.certificate.generation=false', в противном случае аргумент игнорируется.
- -p,--prop-file <arg> - путь к файлу с конфигурационными параметрами. Если этот путь не задан, то используется <lac_dir>/conf/configuration.properties файл.

Требования к паролю

Если механизм аутентификации и авторизации, определенный в `lac.users.auth-type.allowed`, содержит SELF значение, то пароли пользователей зарегистрированных через этот механизм должны удовлетворять определенному набору правил:

- длина пароля не должна быть меньше `lac.users.password.length`;
- все символы пароля должны укладываться в алфавит `lac.users.password.alphabet`;
- в пароле должен присутствовать хотя бы один символ из групп определенных в `lac.users.password.rules`.

Это также касается пароля первоначального административного пользователя ЦУ заданного через аргумент `--admin-password`.

Пример запуска

```
cd <lac_dir>
java -jar ./lib/lac-configurator-1.0.0.jar --database-password password --admin
-password Password1$ --keystore-password password --prop-file  /temp/lac-
configuration.properties
```

Повторный запуск

Конфигуратор работает в системе единожды, он создает схему базы данных, наполняет ее необходимыми для работы значениями, а также создает некоторые файлы необходимые для работы ЦУ.

> **Важно:**
>
> После первого запуска центра управления Libercat и начала работы с управляемыми серверами крайне нежелателен повторный запуск конфигуратора, т.к. повторный запуск приведет к удалению всех созданных ранее данных.

Тем не менее, если пользователь допустил ошибку при создании конфигурации, например, указал неверные данные LDAP групп, у него есть возможность запустить конфигуратор еще раз с исправленными параметрами. Для этого используются следующие аргументы конфигуратора:

- `-m,--mode <arg>` - режим работы конфигуратора. Возможные значения: `configure`, `reconfigure`, `clean`. По умолчанию используется режим `configure`, он используется для первоначального запуска. Режим `clean` используется для очистки деятельности конфигуратора - файловая система и база данных приводится в первоначальное состояние. Режим `reconfigure` используется для повторного запуска конфигуратора с измененными конфигурационными параметрами.
- `-f,--force` - подтверждение удаления ранее созданных данных.

5. Запуск

После того как конфигуратор закончил свою работу, можно запустить центр управления Libercat стандартными способами запуска сервера приложений Libercat EE.

Для Windows системы:

```
<lac_dir>\bin\catalina.bat start
```

Для Unix систем:

```
<lac_dir>/bin/catalina.sh start
```

После запуска сервера центра управления с ним можно начать работать в браузере по адресу https://<lac-init.server.host>:<lac-init.server.front-port>, по умолчанию это будет https://localhost:443.

Остановка или повторный запуск центра управления Libercat осуществляется стандартным способом для сервера приложений Libercat EE. Подробнее эту информацию можно получить из документации на сервера приложений Libercat EE.

6. Эксплуатация

Аутентификация и авторизация пользователей

Для использования центра управления Libercat необходимо аутентифицироваться. Возможные способы аутентификации были определены при запуске конфигуратора параметром `lac.users.auth-type.allowed`. На данный момент поддерживается два способа аутентификации и авторизации:

- внутренний механизм ЦУ Libercat
- внешний LDAP сервис.

Если в конфигурации было определено, что ЦУ поддерживает внутренний механизм аутентификации/авторизации, то после запуска конфигуратора центр управления Libercat будет содержать одного пользователя с ролью **администратор**, определенного параметром `lac-init.users.admin.login`. Для дальнейшей работы с управляемыми серверами необходимо либо расширить роли данного пользователя, либо создать нового(ых) пользователя(ей) с соответствующими ролями.

Если в конфигурации было определено, что ЦУ поддерживает аутентификацию/авторизацию через внешний LDAP сервис, то пользователи создаются автоматически при первичной аутентификацией и имеют роли в соответствии с заданным сопоставлением LDAP групп через `lac-init.users.ldap-config.lac-roles.<LDAP_GROUP_NAME>` параметры.

В конфигурации может быть определено, что доступны оба способа аутентификации/авторизации. Это не означает, что один и тот же пользователь может попеременно аутентифицироваться то одним, то другим способом. Способ аутентификации пользователя определяется его первичной регистрацией: создание пользователя через **администратора** (внутренний механизм аутентификации) или самостоятельный вход через LDAP сервис. Изменить способ аутентификации пользователя после его регистрации невозможно.

При аутентификации зарегистрированного пользователя ограничивается количество неуспешных попыток ввода пароля. Если пользователь сделал некоторое число неудачных попыток, которое соответствуют `lac.users.login.failure-count` параметру, то этот пользователь временно блокируется. У пользователя появится возможность аутентифицироваться в системе с верным паролем только после истечения времени блокировки определенной `lac.users.login.lock-out-time` параметром.

Ролевая модель

Все функции, которые доступны различным пользователям консоли, делятся на функции:

- **наблюдателя** за управляемыми серверами
- **модератора** управляемыми серверами
- **администратора** системы.

Пользователь может обладать одной или более ролями.

Ролевая модель центра управления Libercat выглядит следующий образом:

Рис 3. Ролевая модель центра управления Libercat.

Наблюдатель за управляемыми серверами может получать данные от серверов для мониторинга, но не может вносить никакие изменения в конфигурацию серверов, например, у него нет возможности устанавливать или удалять приложения.

Модератор управляемыми серверами прежде всего имеет права **наблюдателя**, и в дополнение к этому, у него есть возможность изменять конфигурацию серверов, например, устанавливать или удалять приложения *(функции модерации в данной версии ЦУ отсутствуют)*.

Администратору системы доступны функции управления пользователями, например, создание нового пользователя или блокировка существующего пользователя.

Сброс пароля

Возможности сброса пароля существуют только для пользователей, зарегистрированных через встроенный механизм аутентификации. Самостоятельный сброс пароля доступен для любой роли.

Пользователи, зарегистрированные через сторонний LDAP сервис, не могут поменять пароль в ЦУ никаким способом.

При утере пароля

Если пользователь забыл свой пароль, то у него есть возможность сброса пароля через электронную почту, являющуюся его логином. При этом на почту поступит письмо с ссылкой для восстановления пароля. Ссылка будет действительна определенное количество минут заданное параметром `lac.users.password.reset-token.expire` с момента ее генерации.

> **Важно:**
>
> Если конфигурация не содержит настроек почты (параметры `lac.users.notifications.*`), то самостоятельное восстановление пароля в случае его утери будет отсутствовать. Единственной возможностью сбросить пароль будет через ручные действия другого пользователя с ролью *администратора*.

При успешной аутентификации

Пользователь может самостоятельно обновить свой пароль путем ввода текущего и нового пароля. Это может быть как произвольное решение, так и требование системы в момент авторизации. ЦУ не позволяет использование пароля пользователем больше дней, чем определено в `lac.users.password.expire` параметре. Кроме того, пользователь не может задать новый пароль, который будет одинаковым по значению с некоторым количеством последних паролей пользователя определенным `lac.users.password.old-values` параметром.

Администратором

Пользователь с ролью **администратор** может принудительно обновить пароль другому пользователю (см.

подробности ниже).

Управление пользователями

Данная функциональность доступна только пользователям с ролью **администратор** и включает в себя функции редактирования профилей пользователя, а также управлением их доступа к системе посредством определения ролей и блокировок.

Данные о пользователях

Пользователь с ролью **администратор** может получить данные о всех пользователях зарегистрированных в ЦУ в виде таблицы с возможностью фильтрации и сортировки по различным свойствам пользователей. Список данных о пользователях на данный момент ограничен:

- отображаемым именем пользователя;
- ролью в ЦУ, возможные значения: *администратор*, *наблюдатель*, *модератор* и комбинации этих значений;
- логином пользователя;
- типом авторизации, возможные значения: *LDAP*, *стандартный* (т.е. через внутренний механизм аутентификации/авторизации ЦУ);
- статусом пользователя, возможные значения: *активный* , *заблокирован*;
- датой последней смены пароля.

Также у **администратора** есть возможность просмотра подробной информации о каждом отдельно взятом пользователе и изменения его профиля (см. ниже). В карточке профиля пользователя к уже перечисленной информации добавляется **дата последнего входа пользователя в ЦУ**.

Создание нового пользователя

Если в конфигурации было определено, что ЦУ поддерживает внутренний механизм аутентификации/авторизации (`lac.users.auth-type.allowed` содержит значение *SELF*), то у пользователя с ролью **администратор** есть возможность создания новых пользователей ЦУ с аутентификацией через внутренний механизм.

Администратор должен задать новому пользователю логин, имя и его роли в системе. Логин нового пользователя должен соответствовать актуальному электронному адресу пользователя.

Если конфигурация содержит правильные настройки почты (параметры `lac.users.notifications.*`), то новому пользователю должно прийти письмо на электронный адрес с ссылкой для задания первоначального пароля. Ссылка будет действительна определенное количество минут заданное параметром `lac.users.password.reset-token.expire` с момента ее генерации.

Если логин пользователя не соответствует действительному электронному адресу или конфигурация системы не содержит правильные настройки почты (параметры `lac.users.notifications.*`), то задать первичный пароль может только действующий пользователь с ролью *администратор*.

Редактирование пользователя

Администратор может изменить любому пользователю отображаемое имя в центре управления Libercat.

Кроме того, если пользователь аутентифицируется через внутренний механизм аутентификации/авторизации, то администратор может назначить ему одну или несколько новых ролей.

Изменение пароля

Если пользователь аутентифицируется через внутренний механизм аутентификации/авторизации, то администратор может принудительно изменить такому пользователю пароль от системы.

Кроме того, для таких пользователей **администратор** может запросить смену пароля путем посылки пользователю уведомления со ссылкой для задания нового пароля. Эта функция работает, только если конфигурация содержит правильные настройки почты (параметры `lac.users.notifications.*`).

Блокировка/Разблокировка пользователя

На данный момент центр управления Libercat не предполагает функцию удаления пользователей для поддержания аудита и целостности данных.

Пользователи, которые аутентифицируется через внутренний механизм аутентификации/авторизации, могут быть заблокированы (а также разблокированы) администратором. Заблокированные пользователи не будут иметь доступ к системе и не смогут пользоваться никакими ее функциями. Разблокированным пользователям возвращаются все возможности по использованию ЦУ в рамках их ролей.

Доступ пользователей, которые аутентифицируется через внешний сервис LDAP, к системе определяется

вхождением в ту или иную группу LDAP, соответствующую роли(ям) центра управления Libercat (параметры конфигуратора `lac-init.users.ldap-config.lac-roles.<LDAP_GROUP_NAME>`). Для таких пользователей блокировка и разблокировка осуществляется за счет удаления или добавления из/в LDAP группы.

Наблюдение за управляемым серверами

Наблюдение за управляемыми серверами Libercat (EE) - это основополагающая функция центра управления.

Данная функциональность доступна пользователям с ролью *наблюдатель* или *модератор* позволяет отслеживать в реальном времени ключевые характеристики управляемых серверов, такие как список работающих приложений, установленных библиотек, подключенных источников данных и т.д. Количество наблюдаемых характеристик будет расширяться в следующих версиях системы.

Основные данные о серверах

Основные характеристики управляемых серверов представлены в виде таблицы с возможностью фильтрации и сортировки по различным свойствам. В этой таблице также представлена информация о статусе каждого сервера. Статус сервера на данный момент не обновляется автоматически, чтобы увидеть актуальную информацию, необходимо обновить страницу. Список характеристик сервера на данный момент ограничен:

- именем сервера;
- статусом, возможные значения: *доступен, недоступен*, а также временем последней проверки доступности;
- типом сервера, возможные значения: *Libercat, Libercat EE*;
- версией сервера;
- установленной версией Java;
- IP адресом сервера;
- произвольным описанием.

Также у *наблюдателя/модератора* есть возможность просмотра подробной информации о каждом отдельно взятом сервере.

Приложения

Наблюдателю/модератору доступен список приложений, запущенных на управляемом сервере. Основные характеристики приложений представлены в виде таблицы с возможностью фильтрации и сортировки по различным свойствам. Список характеристик приложений на данный момент ограничен:

- контекстным путем приложения;
- именем приложения;
- описанием приложения.

Источники данных

Наблюдателю/модератору доступен список источников данных, сконфигурированных на управляемом сервере. Источники данных могут быть сконфигурированы как средствами Libercat, так и средствами Libercat EE (см. документацию сервера приложений), кроме того, они могут быть сконфигурированы как глобальный ресурс для всех приложений, так и индивидуально для конкретного приложения. Все виды источников данных будут доступны для просмотра наблюдателю/модератору системы.

Основные характеристики источников данных представлены в виде таблицы с возможностью фильтрации и сортировки по различным свойствам. Список характеристик приложений на данный момент ограничен:

- именем источника данных;
- драйвером JDBC;
- JDBC URL;
- именем пользователя.

Библиотеки

Наблюдателю/модератору доступен список библиотек, установленных на управляемом сервере и загруженных **common** или **shared** загрузчиком классов (см. документацию сервера приложений). Основные характеристики библиотек представлены в виде таблицы с возможностью фильтрации и сортировки по различным свойствам. Список характеристик библиотек на данный момент ограничен:

- именем библиотеки;
- полным путем к библиотеке на управляемом сервере;
- типом библиотеки, возможные значения: *системная, пользовательская*. К *системным* относятся библиотеки, включенные в состав сервера приложений Libercat или Libercat EE определенной версии,

а также сервисная(ые) библиотека(и) агента ЦУ Libercat.

- версией библиотеки.

Файлы конфигурации

Наблюдателю/модератору доступен список всех файлов конфигурации с их полными путями используемых на управляемом сервере. Файлы конфигурации можно отфильтровать по имени, а также посмотреть каждый файл в отдельном окне с соответствующей подсветкой синтаксиса.

Журналы

Наблюдателю/модератору доступен список всех файлов-журналов с их полными путями, которые были сохранены на управляемом сервере. Файлы-журналы можно отфильтровать по имени, а также посмотреть каждый файл в отдельном окне.

Количество последних строк для просмотра файл-журнала может быть настроено в окне в зависимости от потребности пользователя. Кроме того, пользователь может скачать полный файл-журнал на свою машину в текстовом виде.

Окружение

Наблюдатель/модератор может ознакомиться со значениями переменных окружения и параметрами виртуальной машины, которые были определены в конфигурации в параметрах `lac.servers.data.system-properties` и `lac.servers.data.environment-variable`. Как переменные окружения, так и параметры виртуальной машины можно фильтровать по имени.

Подключение нового сервера

Это функция доступна только пользователем с ролью *модератор*. Подробная инструкция по подключению управляемого сервера представлена в самом центре управления Libercat на вкладке **Серверы** > **Подключить сервер**. Необходимо последовательно пройти по всем описанным шагам в этой вкладке, а именно:

1. Сначала надо убедиться, что подключаемый сервер приложений это Libercat или Libercat EE и для его работы используется версия Java 8.x и новее.

   Затем необходимо ввести сетевые настройки подключаемого сервера приложений. К обязательным настройкам относятся:

- доменное имя сервера;
- порт агента.

Дополнительный параметр - `bind address`. Он используется на стороне агента ЦУ для поднятия HTTPS сервера для общения с центром управления. Если этот параметр не задан, то будет использован адрес *0.0.0.0*.

2. Агент центра управления на своей стороне поднимает HTTPS сервер для общения с ЦУ, поэтому в рамках подключения управляемого сервера надо задать параметры SSL-конфигурации. Возможны два варианта конфигурации:
   - автоматическая генерация агентского keystore системой, при этом надо учесть, что сертификат в таком keystore будет self-signed. При генерации сертификата используются конфигурационные параметры `lac.certificate.generation.*`. Пароль к keystore будет сгенерирован автоматически системой. Также будет сгенерирован truststore, в котором находится сертификат центра управления. Пароль к truststore также будет сгенерирован автоматически системой.
   - агентский keystore может быть предоставлен пользователем. В этом случае надо указать информацию о существующем агентском keystore:
     - местоположение файла;
     - тип keystore. Возможные значения: *JKS*.
     - пароль к keystore;
     - имя сертификата в keystore.

       Автоматически будет сгенерирован truststore, в котором находится сертификат центра управления, и пароль к нему.
3. Теперь можно скачать агента центра управления Libercat как единый архив в формате **tar.gz** для Unix платформ и в формате **zip** для Windows. Подготовленный архив предназначен именно для того управляемого сервера, который работает на хосте, указанном в шаге **1**, его нельзя использовать на других хостах. Оба варианта архива содержат:
   - агент ЦУ Libercat для управляемых серверов;
   - краткая документация на агент ЦУ;
   - keystore с сертификатом агента и его приватным ключом;
   - truststore с сертификатом ЦУ Libercat.
4. Распакуйте скачанный архив в каталог *CATALINA_BASE* подключаемого сервера приложений (см. документацию сервера приложений).

При правильной распаковке архива в каталоге *CATALINA_BASE* должен появиться файл *lac-agent-README.md*.

5. Затем необходимо сделать следующие шаги в зависимости от операционной системы:

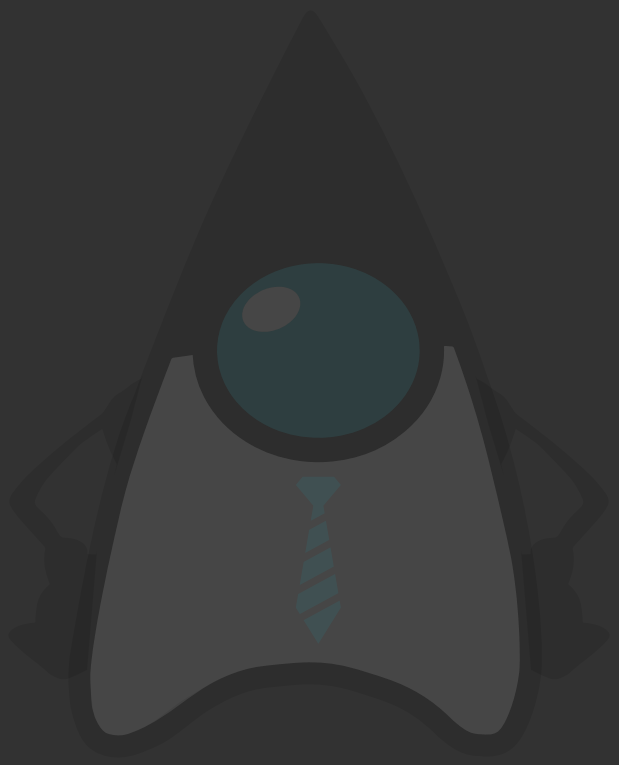
   Для Windows системы

   - если в каталоге <*CATALINA_BASE*>*bin* нет файла *setenv.bat*, создайте пустой файл с таким именем;
   - дополнить файл <*CATALINA_BASE*>*bin**setenv.bat* строками, сгенерированными для этого управляемого сервера в зависимости от настроек, указанных пользователем. Строки приведены в самом шаге.

   Для Unix платформ

   - если в каталоге <*CATALINA_BASE*>*/bin* нет файла *setenv.sh*, создайте пустой файл с таким именем;
   - дополнить файл <*CATALINA_BASE*>*/bin/setenv.sh* строками, сгенерированными для этого управляемого сервера в зависимости от настроек, указанных пользователем. Строки приведены в самом шаге.

6. Перезапустить управляемый сервер и проверить, что он отобразился в списке серверов приложений центра управления.



Центр Управления Libercat
Руководство пользователя

AXIOM JDK