



## SmartTUBE 5. Описание конфигурационных файлов



©2017 ООО «СмартЛабс»

Настоящий документ описывает конфигурационные файлы SmartTUBE 5.

ООО «СмартЛабс» оставляет за собой право на изменение и доработку любых компонентов, описанных в данном документе, а также на изменение самого документа без предварительного уведомления. Следите за изменениями на веб-сайте ООО «СмартЛабс».

SmartTUBE, логотип SmartLabs являются торговыми знаками ООО «СмартЛабс».

Конфиденциально. Ни одна часть данного документа не может быть перепечатана, опубликована, передана другим лицам без специального разрешения от ООО «СмартЛабс».

## Оглавление

1	Введение .....	4
2	Общая часть всех *.conf-файлов .....	5
2.1	Пример .....	5
2.2	Описание параметров .....	5
3	system.conf .....	7
3.1	Пример .....	7
3.2	Описание параметров .....	7
4	container.conf .....	9
4.1	Пример .....	9
4.2	Описание параметров .....	9
5	cache-server.conf .....	11
5.1	Пример .....	11
5.2	Описание параметров .....	11
6	data-model-core.conf .....	12
6.1	Пример .....	12
7	data-server.conf .....	12
7.1	Пример .....	12
7.2	Описание параметров .....	13
8	report.conf .....	14
8.1	Пример .....	14
8.2	Описание параметров .....	15

## 1 Введение

**Конфигурационный файл** - это текстовый файл в формате JSON, содержащий метаданные и настройки компонента сервера SmartTUBE. Конфигурационный файл должен иметь общую часть, одинаковую для всех конфигурационных файлов и часть, актуальную только для конкретного компонента.

Общая часть описана в разделе [Общая часть всех \\*.conf-файлов](#).

Прочие свойства описаны в разделах, посвященных конкретным конфигурационным файлам.

Пример задания свойств (конфигурационный файл для компонента SmartSDPDataServer):

```
properties = {
    sdp-properties-file-path = "/opt/smartlabs/smarttube/master/config/" #
    свойство sdp-properties-file-path имеет строковое значение
    http-request-handlers = 2000 #
    свойство http-request-handlers имеет целочисленное значение
    database = { #
database - комплексное свойство
        server = "postgresql" #
реальное имя свойства будет database.server
        connection-string = "jdbc:postgresql://pgsql_db_ip:5432/psql_dbname"
        login = "login"
        password = "password"
        max-connections = 100 #
реальное имя свойства будет database.max-connections
    }
    scheduled-jobs = [ #
    свойство - массив JSON-объектов
        {
            enabled = false
            runWhen = "0 59 23 * * ?" # every day at 23:59 #
каждый день в 23:59
            method = "cacheoperations#reload"
            params = {
                caches = "MetaContentYear, Epg"
            }
        }
    ]
    trace-http-requests = true #
    свойство trace-http-requests имеет логическое значение
}
```

## 2 Общая часть всех \*.conf-файлов

### 2.1 Пример

```
component = {
    description = "SDP Cache Server"
    jar = "sdp-cache-server.jar"
    class = "smartsdp.cacheserver.main.SmartSDPCacheServer"
    singleton = true
    load-on-startup = 12
    servlet-context-path = "/api/*;/cmds/*"
    install-path = "/opt/smartlabs/sdp/modules/cache-server"
    properties = {
        ...
    }
}
```

### 2.2 Описание параметров

Имя параметра	Описание
install-path	Полный путь к директории, в которой установлен компонент. В папке <i>Component-Install-Directory/bin</i> должен находиться JAR-файл компонента, а в папке <i>Component-Install-Directory/lib</i> - JAR-файлы внешних библиотек, от которых зависит компонент
jar	Имя JAR-файла компонента. Например, <i>SmartBGPTableAggregator.jar</i>
class	Полное (включая имя Java-пакета) имя главного класса компонента. Например, <i>smartlabs.bgptableaggregator.main.SmartBGPTableAggregator</i>
singleton	Флаг, если <i>yes</i> - компонент будет функционировать в единственном экземпляре, если <i>no</i> - таких компонентов может быть несколько
load-on-startup	Целое число, указывающее порядок запуска компонента: компоненты с меньшим значением <i>load-on-startup</i> запускаются раньше компонентов с большим значением
servlet-context-path	Только для компонентов-сервлетов: контекст HTTP запросов
properties	Начальные значения свойств компонента и связи с другими компонентами.  Бывают простые свойства (например, <i>load-on-startup = 12</i> ) и свойства-коллекции (например, <i>sdp-dataservers = [</i>

"10.25.190.11:8084", "10.25.190.12:8084" ] ) . Каждое свойство имеет имя и значение:

- ▶ **Имя свойства** - идентификатор свойства, по которому его значение будет запрашиваться компонентом.
- ▶ **Значение свойства** - значение свойства, которое передается компоненту

## 3 system.conf

### 3.1 Пример

```
system-properties = {
    default-profile-language = "ru"
    default-profile-currency = "EUR"
    user-accounts.store-days = 30
    pending-notifications.store-hours = 24
    media-positions.store-days = 3
    upload-path = "/opt/smartlabs/images/"
    sdp-dataservers = [ "10.25.190.11:8084", "10.25.190.12:8084" ]
    cache-servers = [ "10.25.190.13:8084", "10.25.190.14:8084" ]
    http-proxy = {
        host = ""
        port = 80
    }
    formats = {
        date-incoming = "MM/dd/yyyy"
        date-outgoing = "MM/dd/yyyy"
        time-incoming = "HH:mm:ss"
        time-outgoing = "HH:mm:ss"
        datetime-incoming = "MM/dd/yyyy HH:mm:ss"
        datetime-outgoing = "MM/dd/yyyy HH:mm:ss"
    }
    subscription-timeout=14400000
}
```

### 3.2 Описание параметров

Имя параметра	Описание
default-profile-language	Код языка профиля пользователя по умолчанию
default-profile-currency	Код валюты по умолчанию
user-accounts.store-days	Количество дней хранения данных пользовательского экаунта в кэше
pending-notifications.store-hours	Количество часов хранения извещений для боксов
media-positions.store-days	Количество дней хранения закладок позиций контента в кэше
upload-path	Папка для загрузки картинок
sdp-dataserver	Хост и порт серверов данных. Порт должен совпадать с портом tomcat-attributes.port в container.conf на серверах,

	где установлен компонент SmartSDPDataServer
cache-server	Хост и порт Cache-серверов. Порт должен совпадать с портом tomcat-attributes.port в container.conf на серверах, где установлен компонент SmartSDPCacheServer
http-proxy	<ul style="list-style-type: none"> <li>▶ host – хост прокси-сервера; если пустой, то прокси не используется.</li> <li>▶ port – порт прокси сервера, если используется</li> </ul>
formats	<ul style="list-style-type: none"> <li>▶ date-incoming – формат дат входящих данных;</li> <li>▶ date-outgoing – формат дат исходящих данных;</li> <li>▶ time-incoming – формат времени входящих данных;</li> <li>▶ time-outgoing – формат времени исходящих данных;</li> <li>▶ datetime-incoming – формат даты-времени входящих данных;</li> <li>▶ datetime-outgoing – формат даты-времени исходящих данных</li> </ul>
subscription-timeout	Таймаут после подписки устройства на канал Comet-нотификации, через который сервер закрывает сокет старого канала, мс

## 4 container.conf

### 4.1 Пример

```
component-container = {
    container-name = "CC on Projects"
    cli-enabled = false # при установке контейнера, как
демона Linux, должно иметь значение false
    control-port = 4440
    log-file-path = "/var/log/smartlabs/sdp"
    log-level = "INFO"
    check-dependences = true
    web-interface-enabled = true
    stop-on-component-error = false
    pause-before-notify-tomcat-runned = 5000
    tomcat-attributes {
        port = 8084
        maxConnections = 10000
        maxThreads = 1000
        connectionTimeout = 10000
        socket-soKeepAlive = true # !!! нужен
socket.soKeepAlive
    }
    remote-logging-server = {
        host = "127.0.0.1"
        port = 5556
    }
}
```

### 4.2 Описание параметров

Имя параметра	Описание
container-name	Имя контейнера компонент (КК). Используется для формирования имени логгера для удобства отладки. По умолчанию используется имя хоста, на котором развернут КК.
cli-enabled	Признак возможности управления КК через командную строку или через команды утилиты <code>ssadmin</code> , посылаемые через сокет.  При установке контейнера, как демона Linux, должно иметь значение <i>false</i>
control-port	Порт управления КК, через который принимаются команды от <code>ssadmin</code> . По умолчанию 4440
log-file-path	Директория для log-файла <i>SmartSDPCompContainer.log</i> .

log-level	Минимальный уровень логирования
check-dependences	Признак необходимости проверки при загрузке компонента версий JAR-файлов библиотек: true   false
web-interface-enabled	<p>Признак разрешения web-интерфейса посредством встроенного контейнера сервлетов Apache Tomcat: true   false.</p> <p>Если web-interface-enabled равно true, то будет запускаться контейнер сервлетов Apache Tomcat, который для своей работы будет создавать директорию <i>Contaner_Install_Directory/work</i>. Это требует наличия у запускающего соответствующих прав</p>
stop-on-component-error	Признак необходимости остановки при ошибке создания/инициализации/запуска компонентов. Если false, то указанные ошибки игнорируются (но запись в лог ведется)
pause-before-notify-tomcat-runned	Пауза в мсек, после которой компоненты извещаются об успешном запуске Tomcat
tomcat-attributes.port	Номер порта встроенного Apache Tomcat
tomcat-attributes.maxConnections	Максимальное количество соединений для встроенного Apache Tomcat
tomcat-attributes.maxThreads	Максимальное количество потоков обслуживания для встроенного Apache Tomcat
tomcat-attributes.connectionTimeout	Таймаут подключения Apache Tomcat
tomcat-attributes.socket-soKeepAlive	Значение настройки сокета keep-alive (Apache Tomcat): true/false
remote-logging-server.host	Хост удаленного сервера логирования
remote-logging-server.port	<p>Порт удаленного сервера логирования</p> <p>Выключить удаленное логирование можно полностью удалив раздел remote-logging-server</p>

## 5 cache-server.conf

### 5.1 Пример

```
component = {
  description = "SDP Cache Server"
  jar = "sdp-cache-server.jar"
  class = "smartsdp.cacheserver.main.SmartSDPCacheServer"
  singleton = true
  load-on-startup = 12
  servlet-context-path = "/api*/;/cmds/*"
  install-path = "/opt/smartlabs/sdp/modules/cache-server"
  properties = {
    sdp-properties-file-path = "/opt/smartlabs/smarttube/master/config/"
    elasticsearch = {
      cluster-name = "smart_cache_dev"
      nodes = [ "127.0.0.1:9300" ]
    }
    number-of-http-get-handlers = 2000
    number-of-http-post-handlers = 10
    number-of-notification-handlers = 10
    trace-notification = false
    recreate-indices-on-startup = false
    trace-http-requests = true
  }
}
```

### 5.2 Описание параметров

Имя параметра	Описание
description, jar...	<a href="#">Общая часть всех *.conf-файлов</a>
servlet-context-path	МAPPING на /api и /cmds
sdp-properties-file-path	Путь к файлу настроек сервера SmartTUBE (по умолчанию "/opt/smartlabs/smarttube/master/config/")
elasticsearch.cluster-name	Имя кластера сервера Elasticsearch. Должно быть уникально в локальной сети
elasticsearch.nodes	Сервера Elasticsearch, объединенные в кластер
number-of-http-get-handlers	Количество рабочих потоков, обрабатывающих HTTP GET запросы
number-of-http-post-handlers	Количество рабочих потоков, обрабатывающих HTTP POST запросы
number-of-notification-	Количество обработчиков уведомлений

handlers	
trace-notification	Флаг трассировки нотификаций
recreate-indices-on-startup	Признак необходимости перезагрузки всех кэшей при перезапуске компонента
trace-http-requests	Флаг трассировки в логе HTTP-запросов

## 6 data-model-core.conf

### 6.1 Пример

```

component = {
  description = "Core Data Model"
  jar = "sdp-datamodel-core.jar"
  class = "smartsdp.datamodelcore.main.SmartSDPDataModelCore"
  singleton = true
  load-on-startup = 1
  install-path = "/opt/smartlabs/sdp/modules/data-core"
}

```

Описание параметров см. [Общая часть всех \\*.conf-файлов](#)

Специальных свойств для компоненты нет.

## 7 data-server.conf

### 7.1 Пример

```

component = {
  description = "JSON Data Server"
  jar = "sdp-data-server.jar"
  class = "smartsdp.dataserver.main.SmartSDPDataServer"
  singleton = true
  load-on-startup = 10
  servlet-context-path = "/sdpapi/*;/sdpapibd/*"
  install-path = "/opt/smartlabs/sdp/modules/data-server"
  properties = {
    sdp-properties-file-path = "/opt/smartlabs/smarttube/master/config/"
    http-request-handlers = 2000
    database = {
      server = "postgresql"
      connection-string = "jdbc:postgresql://pgsql_db_ip:5432/psql_dbname"
      login = "login"
      password = "password"
      max-connections = 100
    }
  }
  notifications = {
    database-path = "/var/smartlabs/sdp/data-server/db"
    receptionists = [ "127.0.0.1:5051" ]
    dispatch-jobs-period-sec = 1000
  }
}

```

```

        dispatch-batch-size = 1000
        trace-dispatching = false
    }
    scheduled-jobs = [
        {
            enabled = false
            runWhen = "0 59 23 * * ?" # every day at 23:59
            method = "cacheoperations#reload"
            params = {
                caches = "MetaContentYear, Epg"
            }
        }
    ]
    trace-http-requests = true
}
}

```

## 7.2 Описание параметров

Имя параметра	Описание
description, jar...	<a href="#">Общая часть всех *conf-файлов</a>
sdp-properties-file-path	Путь к файлу настроек сервера SmartTUBE (по умолчанию "/opt/smartlabs/smarttube/master/config/")
http-request-handlers	Количество обработчиков HTTP-запросов
database.server	Тип сервера базы данных (БД)
database.connection-string	Строка соединения с сервером БД
database.login	Логин пользователя БД
database.password	Пароль пользователя БД
database.max-connections	Максимальное количество подключений к БД
notifications.database-path	Путь к папке для извещений БД
notifications.receptionists	
notifications.dispatch-jobs-period-sec	

notifications.dispatch-batch-size	
notifications.trace-dispatching	
scheduled-jobs	<p>JSON-массив объектов, описывающих периодически запускаемые задания. Каждый JSON-объект задания имеет следующий формат:</p> <ul style="list-style-type: none"> <li>▶ name - имя задания.</li> <li>▶ enabled - флаг разрешения запуска данного задания.</li> <li>▶ runWhen - время/период запуска задания (строка). Описание формата см. по ссылке: <a href="http://www.quartz-scheduler.org/documentation/quartz-2.x/tutorials/tutorial-lesson-06.html">http://www.quartz-scheduler.org/documentation/quartz-2.x/tutorials/tutorial-lesson-06.html</a>.</li> <li>▶ method - имя сервиса и метода для запуска в формате "сервис#метод".</li> <li>▶ params - параметры, передаваемые методу при запуске. Например: <ul style="list-style-type: none"> <li><u>для импорта EPG из файла</u> epg-import.url - URL файла с EPG.</li> <li>epg-import.login - логин для доступа к файлу EPG.</li> <li>epg-import.password - пароль для доступа к файлу EPG.</li> <li><u>для импорта прогноза погоды из внешней системы</u> appid - идентификатор приложения, из которого будет импортироваться прогноз</li> </ul> </li> </ul>
trace-http-requests	Флаг трассировки в логе HTTP-запросов

## 8 report.conf

### 8.1 Пример

```

component = {
  description = "SDP Report"
  jar = "sdp-report.jar"
  class = "smartsdp.report.main.SmartSDPReport"
  singleton = true
  load-on-startup = 14
  servlet-context-path = "/report/*"

```

```
install-path = "/opt/smartlabs/sdp/modules/reports"
#install-path = "/SmartSDPReport/target"
properties = {
    sdp-properties-file-path = "/opt/smartlabs/smarttube/master/config/"
    number-of-http-get-handlers = 100
}
}
```

## 8.2 Описание параметров

Имя параметра	Описание
description, jar...	<a href="#">Общая часть всех *conf-файлов</a>
sdp-properties-file-path	Путь к файлу настроек сервера SmartTUBE (по умолчанию "/opt/smartlabs/smarttube/master/config/")
number-of-http-get-handlers	Количество обработчиков HTTP GET запросов