Загрузка по протоколу REST (КС)

Управление сервисом загрузки справочников на базе **Ubuntu 18.04** осуществляется следующим образом:

1. Для сервиса необходимо вручную создать пустой файл /etc/default/exchangers-rest-upload:

   ```
   touch /etc/default/exchangers-rest-upload
   ```

2. Запустить сервис с помощью команд:

   ```
   systemctl start exchangers-rest-upload
   ```

3. Посмотреть статус сервиса можно командой:

   ```
   systemctl status exchangers-rest-upload
   ```

   После старта сервис будет работать даже после перезагрузки.

Настройка обмена через REST

Настройка сервиса обмена REST осуществляется в конфигурационном файле /opt/exchangers/config/cs/exchangers.ini в секции [REST_UPLOAD].

Наименование	Тип данных	Возможные значения	Описание	Примечания
port	целочисленный	целое положительное число	Порт для REST сервиса	По умолчанию 28080
authorize	логический	• true • false	Проверять логин и пароль при REST запросе	По умолчанию true
login	строковый		Логин для авторизации REST запроса	По умолчанию admin. Используется для аутентификации сообщений
passwd	строковый		Пароль для авторизации REST запроса	По умолчанию admin. Используется для аутентификации сообщений
shopCodeIsShopId	логический	• true • false	Являются ли коды магазинов, которые передают в запросе, идентификаторами магазинов	По умолчанию true
maxCountBackupRequests	целочисленный	целое положительное число	Максимальное количество архивов запросов для одного магазина в бэкап директории	Устаревшие архивы удаляются. При значении 0 настройка не учитывается
actualCountDaysForBackupRequests	целочисленный	целое положительное число	Количество дней актуальности архивов запросов для одного магазина в бэкап директории	Устаревшие архивы удаляются. При значении 0 настройка не учитывается
sessionIdleTimeout	целочисленный	целое положительное число	Время простоя сессии (в секундах)	По умолчанию - 300 сек. Если истекло время, указанное в данной настройке, после последнего запроса внутри сессии, то вся сессия отменится
useSession	логический	• true • false	Использовать сессионную загрузку справочников	По умолчанию false
singleShopSession	логический	• true • false	В один момент времени разрешать грузить только один справочник в магазин при использовании сессионного режима	По умолчанию false

Пример настройки

```
[REST_UPLOAD]
; Порт для REST сервиса
port = 28080
; Проверять логин и пароль при REST запросе. По умолчанию - True
authorize = True
; Логин для авторизации REST запроса. По умолчанию - admin
login = admin
; Пароль для авторизации REST запроса. По умолчанию - admin
passwd = admin
; Являются ли коды магазинов, которые передают в запросе идентификаторами магазинов
shopCodeIsShopId = True
; Максимальное количество архивов запросов для одного магазина в бэкап директории (устаревшие архивы будут удаляться)
; Если указать значение 0, то данная настройка учитываться не будет
maxCountBackupRequests = 0
; Количество дней актуальности архивов запросов для одного магазина в бэкап директории (устаревшие архивы будут удаляться)
; Если указать значение 0, то данная настройка учитываться не будет
actualCountDaysForBackupRequests = 3
; Время простоя сессии (в секундах). Если истекло время, указанное в данной настройке,
; После последнего запроса внутри сессии, то вся сессия отменится. По умолчанию - 300 сек
sessionIdleTimeout = 300
; Использовать сессионную загрузку справочников. По умолчанию - False
useSession = False
; В один момент времени разрешать грузить только один справочник в магазин при использовании сессионного режима
singleShopSession = False
```

Архивы запросов на загрузку справочников хранятся помагазинно в директории /opt/exchangers/logs/REST_BACKUP. Количество архивных запросов и количество дней, в течение которых запросы остаются актуальными, определяется параметрами maxCountBackupRequests и actualCountDaysForBackupRequests соответственно. Устаревшие запросы удаляются. Если параметры не определены, то в директории хранятся все переданные запросы.

Загрузка справочников

Загрузка справочников по REST протоколу осуществляется посредством запросов типа POST на URL:

```
http://<host сервера>:<порт REST сервиса>/aifexchange/loaddicts?shopcode=<код магазина>&dictionaryid=<идентификатор справочника>&addressstatusserver=<хост:порт>
```

В параметрах запроса могут быть переданы следующие данные:

Наименование	Тип данных	Описание	Примечания
shopcode	строковый	Код магазина, в БД которого будет загружен справочник	Если поле не заполнено, то данные загружаются в БД All. Если в поле указан код категории, то данные будут загружены в БД Категории
dictionaryid	строковый	Идентификатор справочника	Не является обязательным параметром. Требуется для отслеживания загрузки справочника. Если не указан, то генерируется уникальный внутренний идентификатор
addressstatusserver	строковый	Адрес и порт сервиса отслеживания статусов загрузки	Если адрес указан, то на него приходят сообщения от сервисов Exchanger и Nes. Если поле не заполнено, то статусы загрузки не отправляются

Запросы и ответы, передаваемые и получаемые по REST протоколу, представляют собой массив AIF-объектов в формате json.

Пример массива объектов

```
[{
"command":"addUnit",
"unit":{
      "unitCode":796,
      "name":"шт",
      "fractional":false
      }
},
{
"command":"addDept",
"dept":{
      "code":2,
      "name":"ИП Кузнецов С.В.",
      "documentPrefix":"К"
      }
},
{"command": "clearCard"},
{
"command":"addCard",
"card":{
      "idcard":"7951380000001",
      "number":"7951380000001",
      "idcardgroup":1
      }
}
]
```

Пример генерации запроса

```
import httplib2
import base64
import json
import time
dictionaryId = str(time.time())
print 'dictionaryId = %s' % dictionaryId
shopcode = "111"
serverHost = 'localhost'
serverHttpPort = 28080
login = 'login'
password = 'passw'
http = httplib2.Http(".cache")
base64string = base64.encodestring('%s:%s' % (login, password))[:-1]
authheader = "Basic %s" % base64string
jsonHeader = {'Content-Type':'application/json', "Accept": "application/json", "Authorization": authheader}
method = 'POST'
addressStatusServer='192.168.0.1:18080'
url = 'http://%s:%s/aifexchange/loaddicts?shopcode=%s&dictionaryid=%s&addressstatusserver=%s' % (serverHost, serverHttpPort, shopcode, dictionaryId,
addressStatusServer)

countRows = 10
dictionary = []
for i in range(countRows):
command = {"command":"addInventItem", "invent":{ "deptCode":1, "measureCode":"796", "minPrice":0, "inventcode":"0000000007%s" % i, "isInventItem":True,
"price":130+i,
"taxGroupCode":2, "isInvent":True, "discAutoScheme":0, "barcode":"8718343465%s" % i, "extendedOptions":"extendedOptions", "rtext":"rtext %s" % i, "name":"
Name %s" % i } }
dictionary.append(command)

body = json.dumps(dictionary)
response, content = http.request(url, method, headers=jsonHeader, body=body)
if response.status == 200:
print json.loads(content)
else:
print response, content
```

Пример ответа

```
{"success": true, "info": "OK"}
```

, где
success - статус обработки справочника,
info - дополнительная информация.
В случае ошибки в этом поле будет сообщение об ошибке.

Для получения статусов загрузки справочников может быть настроен сервис получения статусов:

Пример REST-сервиса получения статусов

```
from bottle import route, run, response, request
from httplib import INTERNAL_SERVER_ERROR
import time
import traceback
import json
STATUS_ERROR = 'STATUS_ERROR'                    # Ошибка загрузки/выгрузки справочника
STATUS_READY = 'STATUS_READY'                    # Справочник успешно загружен/выгружен
STATUS_READY_WITH_ERROR = 'STATUS_READY_WITH_ERROR' # Справочник загружен с ошибками
STATUS_DURING = 'STATUS_DURING'                  # Справочник в данный момент загружается/выгружается

results = {'exchanger': 0, 'nes': 0} # -1 - error, 0 - during, 1 - OK
def setResult(status, typeStatus):
    global results
    if status:
        if status in [STATUS_ERROR, STATUS_READY_WITH_ERROR]:
            results[typeStatus] = -1
        elif status == STATUS_READY:
            results[typeStatus] = 1

def restMethod(result, typeStatus):
    response.content_type = 'application/json'
    try:
        if (request.body.len > 0):
            body = json.loads(request.body.read(), encoding='utf8')
            status = body.get('status', None)
            print('%s status: %s' % (typeStatus, status))
            setResult(status, typeStatus)
    except Exception, e:
        print(traceback.format_exc())
        response.status = INTERNAL_SERVER_ERROR
    return json.dumps(result, ensure_ascii=False)
@route('/CSrest/rest/dicts/backOfficeState/:shopCode', method='POST')
def restStatus(shopCode):
    result = str(time.time())
    return restMethod(result, 'exchanger')
@route('/CSrest/rest/dicts/backOfficeState/:shopCode/:dictStateId', method='PUT')
def restStatus(shopCode, dictStateId):
    result = dictStateId
    return restMethod(result, 'exchanger')
@route('/CSrest/rest/dicts/cashLoad/:shopCode/:cashCode/:dictStateId', method='PUT')
def restLoadStatus(shopCode, cashCode, dictStateId):
    result = dictStateId
    return restMethod(result, 'nes')
def runRest():
    run(host='192.168.1.1', port='18080')
if __name__ == '__main__':
    runRest()
```

Если настроен сервис отслеживания статусов загрузки, то на него будут приходить ответы:

Пример ответа о статусе загрузки

```
exchanger status: STATUS_DURING
192.169.10.80 - - [17/Sep/2014 10:11:46] "POST /CSrest/rest/dicts/backOfficeState/111 HTTP/1.1" 200 15
exchanger status: STATUS_READY
192.169.10.80 - - [17/Sep/2014 10:11:56] "PUT /CSrest/rest/dicts/backOfficeState/111/1410923501.03 HTTP/1.1" 200 15
nes status: STATUS_DURING
192.169.10.80 - - [17/Sep/2014 10:12:06] "PUT /CSrest/rest/dicts/cashLoad/111/111/1410923501.03 HTTP/1.1" 200 15
nes status: STATUS_READY
192.169.10.80 - - [17/Sep/2014 10:12:16] "PUT /CSrest/rest/dicts/cashLoad/111/111/1410923501.03 HTTP/1.1" 200 15
```

Сессионная загрузка справочников

Загрузку одного справочника несколькими запросами можно выполнить единой порцией данных при помощи сессионной загрузки справочников. Возможность сессионной загрузки справочников определяется параметром useSession. Загрузка в один момент времени только одного справочника в один магазин задается параметром singleShopSession.

Если сессионная загрузка справочников разрешена (useSession = true), то REST-контроллер переходит в режим загрузки справочников порциями, которыми пользователь управляет с помощью REST-запросов типа POST на URL.

Для управления сессиями загрузки используются следующие методы:

- start – запуск сессии.

  Пример REST-запроса

  http://<хост сервера>:<порт REST сервиса>/aifexchange/loaddicts/session/start?shopcode=<код магазина>&dictionaryid=<идентификатор справочника>&addressstatusserver=<хост:порт>

- stop – завершение сессии. Метод должен быть вызван обязательно после окончания отправки справочников, чтобы все переданные части справочника были загружены в БД.

  Пример REST-запроса

  http://<хост сервера>:<порт REST сервиса>/aifexchange/loaddicts/session/stop?shopcode=<код магазина>&dictionaryid=<идентификатор справочника>

- rollback – отмена загрузки справочников. Если сессия была открыта, но за время простоя, определяемого параметром sessionIdleTimeout, не было выполнено ни одного вызова, то сессия отменяется со всеми переданными данными.

  Пример REST-запроса

  http://<хост сервера>:<порт REST сервиса>/aifexchange/loaddicts/session/rollback?shopcode=<код магазина>&dictionaryid=<идентификатор справочника>

В рамках одной сессии может выполняться загрузка данных только на один магазин.

Пример

Для добавления одного справочника товаров, разбитого на 2 запроса, необходимо выполнить следующие действия по порядку:

1. Отправить REST-запрос на запуск сессии.
2. Отправить запрос на загрузку справочника.
3. Отправить запрос на загрузку справочника.
4. Отправить REST-запрос на завершение сессии.

Если сессионная загрузка справочников запрещена (useSession = false), то использование методов управления сессиями будет недоступно. При загрузке справочников одному запросу добавления справочника будет соответствовать один справочник.