RESTup. RESTful java-сервер консольных приложений. Версия 1.3

Rev 61110

rest up:
to relax in order to have strength for something
Cambridge Dictionary

1. Назначение

RESTup - pure java6 сервер, обеспечивает RESTful WEB API к консольным приложениям операционной системы (далее сервисы).

Взаимодействие с сервером производится по следующей общей схеме:
- получить список сервисов (GET), определить URI сервиса;
- создать задание сервису (POST), получить URI для файлов задания;
- передать файл(ы) задания (PUT);
- исполнить параметризованное задание (POST), получить URI файлов-результатов;
- получить список файлов-результатов (GET);
- получить файл(ы) результата (GET);
- удалить задание и связанные файлы (DELETE).

Сервер имеет экспериментальный пользовательский интерфейс (UI) на основе протокола WebDAV.

2. Настройка сервера

Конфигурация сервера хранится в файле RESTupConfig.xml, который берется при запуске из текущего каталога. Имена атрибутов регистрозависимы. Значения атрибутов и их взаимосвязь не контролируются. Значения атрибутов spoolDir, jobCommand зависят от среды исполнения (Linux/ Windows). Ниже приведен пример для Windows-платформы:

```
<?xml version="1.0" encoding="Windows-1251" ?>
<server port="8080" maxJobsStarted="4" jobsLifeTime="240" debugLevel="0">
<service name="Echo"
jobCommand="CMD /C xcopy %inFilesDir%%jobParams% %outFilesDir% /E/Y/Q"
fileExts="" debug="off" jobDefaults="*.*" jobQuota="500000" commandTimeout="10">
Эхо-сервис. Возвращает файл(ы) задания по маске, определяемой параметром задания.
</service>
</server>
```

Параметры сервера (в скобках приведены значения по-умолчаню):

port - порт листенера (80). Листенер прослушивает все доступные интерфейсы.
port="1935", в случае переадресации порта Linux-командой:
iptables -t nat -A PREROUTING -p tcp --dport **80** -j REDIRECT --to **1935**

spoolDir - каталог файлов заданий (подкаталог restup_spool каталога временных файлов системы). Избегайте пробелов в полном пути к каталогу заданий!

jobsLifeTime - время жизни заданий с момента создания (240) сек. По истечении указанного времени задание и связанные файлы удаляются.

maxJobsStarted - максимальное количество исполняемых внешних программ (2)
Если количество исполняемых внешних программ превышает допустимое, задание ставится в очередь на время:
jobsLifeTime-commandTimeout-<время с момента создания>

по истечении которого удаляется.

debugLevel - детализация отладочной информации, выводимой на консоль 0|1|2 (1)

Параметры сервиса:

name - уникальное имя сервиса (обязательно)

fileExts - разрешенные расширения файлов, разделенные запятой (любое, включая создание подкаталогов)

debug - вывод на консоль отладочной информации (off): on|off

jobQuota - максимальный размер файлов задания (без ограничений) байт

jobCommand - исполняемая внешняя команда (обязательно) В команде используются макроподстановки:

%inFilesDir% - полный путь к каталогу файлов задания

%outFilesDir% - полный путь к каталогу файлов-результатов (пути с завершающим разделителем)

%jobParams% - пользовательские параметры задания

jobDefaults - параметры задания по-умолчанию (нет)

commandTimeout - предельное время исполнения внешней программы задания (60) сек.

аннотация сервиса

3. Установка и запуск сервера

3.1 Для запуска сервера требуется JavaSE jre или openJDK runtime 1.6 и старше. Установка сервера заключается в настройке конфигурационного файла.

Конфигурационные файлы содержат примеры сервисов, базирующиеся на свободно распространяемом программном обеспечении, которое, в свою очередь, должно быть предварительно установлено в каталоги 'по-умолчанию':

- LibreOffice 4.2: http://ru.libreoffice.org/;
- Tesseract-OCR: https://code.google.com/p/tesseract-ocr/;
- CheckPFR + архиватор 7-zip: http://www.pfrf.ru/ot_bashkor/administrirovanie/19137.html http://www.7-zip.org/

Сервис	Реализуемое действие	Linux	Windows
Office2pdf	Преобразование офисных документов в формат Adobe PDF	x	x
Office2html	Преобразование офисных документов в формат html	x	x
Office2ooxml	Преобразование офисных документов в формат MS Office 2007 (OOXML)	x	x
Office2mso97	Преобразование офисных документов в формат MS Office 97/2000	x	x
Tesseract-OCR	Оптическое распознавание текста (OCR)	x	x
CheckPFR	Форматно-логический контроль обменных файлов ПФР	-	x
Echo	Тестовый сервис. Возвращает файл(ы) задания по маске, определяемой параметром	x	x

Конфигурационные файлы настроены на запуск скриптов из текущего (общего с конфигурационным файлом) каталога.

3.2 Запуск сервера

java [-Dключ=значение] -jar [путь]RESTupServer.jar

Ключи и значения по-умолчанию:
consoleEncoding=utf-8 - кодировка вывода на консоль.
В среде Windows используется -DconsoleEncoding=cp866.
davEnable=no - yes|no разрешает/запрещает WebDAV интерфейс.

4. RESTful API

Реализует перечисленные в п. 1 действия. Параметры передаются в uri, полях заголовка и теле запроса. Значения возвращаются в полях заголовка и теле ответа. Обмен с сервером производится в кодировке UTF-8. Возвращаемые коды успешного завершения: 200, 201, 204.

Если в заголовке запроса клиента отсутствует поле Host, возвращается код ошибки 400 (Bad Request).

4.1 Получить список сервисов

Запрос клиента:

```
GET /restup/ HTTP/1.1
Host: localhost:8080
Accept: text/xml
```

Ответ сервера:

```
HTTP/1.1 200 OK
Server: RESTup/1.3.xxxx
Connection: close
Content-Type: text/xml; charset=utf-8
Content-Length: xxxxx

<?xml version=”1.0” encoding=”utf-8”?>
<restup>
   <service>
     <uri>http://localhost:8080/restup/echo/</uri>
     <name>Echo</name>
     <fileExts/>
     <jobQuota>500000</jobQuota>
     <jobDefaults>*.*</jobDefaults>
     <abstract>
     Эхо-сервис. Возвращает файл(ы) задания по маске, определяемой параметром
   задания.
     </abstract>
   </service>
…
</restup>
```

4.2 Создать задание сервису, получить URI для файлов задания.

Запрос клиента:

```
POST /restup/echo/ HTTP/1.1
```

```
Host: localhost:8080
Content-Length: 0
```

Ответ сервера:

```
HTTP/1.1 201 Created
...
Location: http://localhost:8080/restup/echo/add03ead02c9bec8/in
Content-Length: 0
```

4.3 Передать файл задания

Запрос клиента:

```
PUT /restup/echo/add03ead02c9bec8/in/phototest.tif HTTP/1.1
Host: localhost:8080
Content-Type: application/octet-stream
Content-Length: 12345

в теле запроса бинарное содержимое файла
```

Ответ сервера:

```
HTTP/1.1 204 No Content
…
Content-Length: 0
```

4.4 Выполнить задание, получить URI файлов-результатов, удалить файлы задания.

Запрос клиента:

```
POST /restup/echo/add03ead02c9bec8/in HTTP/1.1
Host: localhost:8080
Content-Type: text/plain; charset=utf-8
Content-Length: xxxx

в теле запроса может быть указана строка пользовательских параметров задания
```

Ответ сервера:

```
HTTP/1.1 201 Created
...
Location: http://localhost:8080/restup/echo/add03ead02c9bec8/out/
Content-Length: 0
```

4.5 Получить список файлов-результатов задания

Запрос клиента:

```
GET /restup/echo/add03ead02c9bec8/out/ HTTP/1.1
Host: localhost:8080
Accept: text/xml
Content-Length: 0
```

Ответ сервера:

```
HTTP/1.1 200 OK
...
Content-Location: http://localhost:8080/restup/echo/add03ead02c9bec8/out/
```

```
Content-Type: text/xml; charset=”utf-8”
Content-Length: xxxxx

<?xml version=”1.0” encoding=”utf-8”?>
<restup_out>
   <file>
     <name>phototest.tif</name>
     <size>12345</size>
   </file>
   ...
</restup_out>
```

4.6 Получить файл результата

Запрос клиента:

```
GET /restup/echo/add03ead02c9bec8/out/phototest.tif HTTP/1.1
Host: localhost:8080
Content-Length: 0
```

Ответ сервера:

```
HTTP/1.1 200 OK
...
Content-Type: application/octet-stream
Content-Length: xxxxx

в теле ответа бинарное содержимое файла-результата
```

4.7 Удалить задание и связанные файлы

Запрос клиента:

```
DELETE /restup/echo/add03ead02c9bec8/ HTTP/1.1
Host: localhost:8080
Content-Length: 0
```

Ответ сервера:

```
HTTP/1.1 204 No Content
...
Content-Length: 0
```

5. Пользовательский интерфейс (эксперимент)

Пользовательский интерфейс основан на протоколе WebDAV класса 1. Представляет собой набор удаленных виртуальных папок. Вид действия над пользовательскими файлами определяется назначенным сервисом. Информация о подключенных сервисах и ограничениях пользовательской сессии находится в файле справки корневой папки сервера.

ВАЖНО: в данной версии пользователь идентифицируется IP или именем хоста или комбинацией значений заголовков запроса X-Forwarded-For + Via.

5.1 Настройка WebDAV интерфейса сервера.

Настройки интерфейса хранятся в соответствующей секции

конфигурационного файла сервера:

```
<server ...>
<service .../>
...
<davInterface sessionTimeout=“240” sessionQuota=”100000”
helpFileTemplate=”.\Справка.txt”>
<folder uri=”/Test/Echo” serviceName=”Echo” jobParams=””>
Отладочный эхо-сервис.
</folder>
...
</davInterface>
</server>
```

Параметры интерфейса:

sessionTimeout	- таймаут сессии секунд (=jobsLifeTime)
sessionQuota	- ограничение совокупного размера файлов байт (2 ГиБ)
inFolderName	- имя папки файлов заданий (“in”)
outFolderName	- имя папки результатов (“out”)
helpFileTemplate	- текстовый (utf-8+BOM) файл шаблона справки (встроенный)

Параметры папок преобразований:

uri	- уникальный относительный uri (обязательно). Соответствует папке преобразования. Избегайте пробелов в uri!
serviceName	- имя назначенного сервиса. Если не указано, папка доступна только для чтения.
jobParams	- параметры задания (зависят от сервиса)

информация пользователю

5.2 Подключение к серверу

5.2.1 Windows XP, Vista, 7, 8, 10
Проводник:
Боковое меню - Сеть - Подключить сетевой диск
Выбрать диск, в поле адрес ввести: \\host[:port]\restup\dav

Командная строка:
net use * \\host[:port]\restup\dav

Windows XP, Vista не допускают переопределение порта 80.

5.2.2 Linux Gnome (Nautilus)
В главном меню выбрать 'Файл - Подключиться к серверу...'.
В поле 'Адрес сервера' ввести:
dav://host:port/restup/dav

5.2.3 Linux KDE (Dolphin, Konqueror)
В адресной строке ввести:
webdav://host:port/restup/dav

5.2.4 Монтирование davfs

Ubuntu:

```
$ sudo mount -t davfs -o rw,guid=users http://<host>:<port>/restup/dav
<mount_point>
```

openSUSE:

```
$ wdfs http://<host>:<port>/restup/dav <mount_point> -o umask=0770
```

ВАЖНО: проводники и файловые менеджеры кэшируют содержимое удаленных папок. В некоторых случаях необходимо принудительное ручное обновление.

6 Агенты

Агенты обеспечивают программный интерфейс (API) с сервером консольных приложений.

6.1 Oracle PL/SQL API. Пакет RESTUP_AGENT

Пакет restup_agent является надстройкой над Oracle apex_web_service API. Пакет совместим с Oracle-XE.

Перед использованием пакета, администратором Oracle должен быть разрешен доступ к RESTup–серверу (см. DBMS_NETWORK_ACL_ADMIN).

6.1 Типы данных и переменные.

URL RESTup сервера по-умолчанию:

```
default_url constant varchar2(512) := 'http://localhost:8080/restup/';
```

Подтип tp_urlencoded введен для отличия urlencoded строк идентификатора ресурса (URI) от строк символов :

```
subType tp_urlencoded is varchar2(512);  -- urlencoded uri
```

Запись и таблица характеристик сервиса:

```
Type tp_svc_rec is record
( uri tp_urlencoded           -- URI сервиса
, name varchar2(256)          -- имя сервиса
, fileexts varchar2(256)      -- расширения файлов, поддерживаемые сервисом
, jobquota number             -- максимальный размер задания (байт)
);
Type tp_svcs_tbl is table of tp_svc_rec;
```

Запись и таблица файлов результата исполнения задания:

```
Type tp_file_rec is record
( uri tp_urlencoded           -- URI файла
, filename varchar2(256)      -- имя (и расширение) файла
, filesize number             -- размер в байтах
, content BLOB                -- бинарное содержимое файла
);
Type tp_files_tbl is table of tp_file_rec;
```

6.1.2 Обзор подпрограмм пакета

```
services_tbl function   получить таблицу сервисов
service_get function    получить uri сервиса
job_create function     создать задание сервису, вернуть URI файлов задания
jobfile_put procedure   передать файл задания
job_execute function    выполнить задание
resfiles_tbl function   получить таблицу файлов-результатов задания
resfile_get function    получить бинарное содержимое файла-результата
job_delete procedure    удалить задание и связанные файлы
service_execute function      выполнить сервис с одиночным файлом задания
set_response_error_check procedure  разрешить exception (код состояния != 2xx)
```

services_tbl получить таблицу сервисов:

```
function services_tbl
( p_server varchar2 := default_url  --  URL сервера
) return tp_svcs_tbl pipelined;     --  таблица сервисов
```

service_get получить uri сервиса

```
function service_get
( p_service varchar2                -- имя сервиса
, p_server varchar2 := default_url  -- URL сервера
) return tp_urlencoded;             -- URI сервиса
```

job_create создать задание сервису, вернуть URI файлов задания:

```
function job_create
( p_serviceuri tp_urlencoded  --  URI сервиса
) return tp_urlencoded;       --  URI задания
```

job_delete удалить задание и связанные файлы:

```
procedure job_delete
( p_joburi tp_urlencoded      --  URI задания
);
```

jobfile_put передать файл задания (количество файлов ограничено их суммарным размером - параметр jobmaxsize сервиса):

```
procedure jobfile_put
( p_job  tp_urlencoded        --  URI задания
, p_filename varchar2         --  имя файла
, p_content BLOB              --  бинарное содержимое файла
);
```

job_execute выполнить задание с указанным параметром , получить URI файлов-результатов:

```
function job_run
( p_job  tp_urlencoded        --  URI задания
, p_jobparams varchar2 := ''  --  строка параметров задания (зависит от сервиса)
) return tp_urlencoded;       --  URI результатов
```

resfiles_tbl получить таблицу файлов-результатов выполнения задания (без бинарного содержимого файлов):

```
function resfiles_tbl
```

```
( p_uri tp_urlencoded               --  URI результатов
) return tp_files_tbl pipelined;    --  таблица файлов-результатов
```

resfile_get получить бинарное содержимое файла-результата задания:

```
function resfile_get
( p_fileuri tp_urlencoded     --  URI файла
) return BLOB;                --  бинарное содержимое файла
```

service_execute выполнить сервис с одиночным файлом задания, получить коллекцию файлов-результатов задания (включая бинарное содержимое):

```
function service_execute
( p_service varchar2                -- имя сервиса
, p_filename varchar2 := ''         -- имя файла
, p_content  BLOB := null           -- бинарное содержимое файла
, p_jobparams varchar2 := ''        -- параметры задания (зависят от сервиса)
, p_server varchar2 := url_default  -- URL сервера
) return tp_files_tbl;              -- коллекция файлов-результатов
```

Контроль исполнения REST запросов производится с помощью процедур:

```
apex_web_service.g_status_code      -- возвращает код состояния запроса:
      200,201,204 – успех; 400-599 - ошибка

utl_http.set_response_error_check(true)  -- разрешает exception при получении
      кода состояния отличного от 2xx (не поддерживается в Oracle-XE, но можно
      использовать одноименную функцию этого пакета)
```

6.1.3 Пример использования пакета restup_agent

```
declare
  l_uri varchar2(500);-- возвращаемый URI
  l_sblob BLOB; -- исходный файл
  l_rblob BLOB; -- возвращенный файл
-- генерация произвольного бинарного файла
function generateBLOB(p_size number) return BLOB
is
  l_blob BLOB;
begin
  dbms_lob.createtemporary(l_blob,true,dbms_lob.call);
  while dbms_lob.getlength(l_blob) < p_size loop
    dbms_lob.append(l_blob,utl_raw.cast_from_number(dbms_random.value()));
  end loop;
  dbms_lob.trim(l_blob, p_size);
  return l_blob;
end;
begin
-- получить URI эхо-сервиса, вывести код состояния и URI
  l_uri := restup_agent.service_get('echo'
    , 'http://localhost:8080/restup/');
  dbms_output.put_line(apex_web_service.g_status_code||' '||l_uri);
-- создать задание сервису, вывести код состояния и URI файлов задания
  l_uri := restup_agent.job_create(l_uri);
  dbms_output.put_line(apex_web_service.g_status_code||' '||l_uri);
-- сгенерировать содержимое файла и передать с разными расширениями
  l_sblob := generateBLOB(123456);
  restup_agent.jobfile_put(l_uri,'Тест-Файл 1.bin',l_sblob);
  restup_agent.jobfile_put(l_uri,'Тест-Файл 2.tmp',l_sblob);
-- выполнить задание, указав маску возвращаемых файлов
-- вывести код состояния и URI файлов результата
```

```
  l_uri:=restup_agent.job_execute(l_uri,'*.tmp');
  dbms_output.put_line(apex_web_service.g_status_code||' '||l_uri);
-- пролистать список файлов-результатов исполненного задания
  for t in (select * from table(restup_agent.resfiles_tbl(l_uri))) loop
    dbms_output.put_line(t.filename||' '||t.filesize);
-- получить файл-результата
    l_rblob:=restup_agent.resfile_get(t.uri);
-- сравнить исходный и полученный файлы
    dbms_output.put_line(apex_web_service.g_status_code||' '
      ||dbms_lob.compare(l_sblob,l_rblob));
    if dbms_lob.istemporary(l_rblob)=1 -- освободить LOB
      then dbms_lob.freetemporary(l_rblob); end if;
  end loop;
  restup_agent.job_delete(l_uri); -- удалить задание
  if dbms_lob.istemporary(l_sblob)=1   -- освободить LOB
    then dbms_lob.freetemporary(l_sblob); end if;
end;
```

Результат исполнения (Oracle-XE 11g):

```
200 http://server:8080/restup/echo/
201 http://server:8080/restup/echo/905a69ab9313dae6/in/
201 http://server:8080/restup/echo/905a69ab9313dae6/out/
Тест-Файл 2.tmp 123456
200 0

Statement processed.

0.97 seconds
```

6.2 Java агент. Пакет org.net.restupAgent

Пакет поставляется в исходных текстах. Документация в формате javadoc доступна после генерации. Пример использования:

```
import org.net.restupAgent.*;

import java.io.File;
/**
 * Простой java RESTup client
 */
public class RESTupClient {
  public static void main(String[] args) throws Throwable {
    System.out.println("");
    if (args.length != 1 && args.length !=5 ) {
      System.out.println("Использование:"
        + "\njava -jar RESTupClient.jar <server>"
        + "\n   получить список сервисов"
        + "\njava -jar RESTupClient.jar"
        + "<server> <service> <servicePrm> <jobFile> <resultDir>"
        + "\n   исполнить удаленный сервис. Пример:"
        + "\njava -jar RESTupClient.jar \"http://localhost:8080\""
        + " echo \"\" \"./org\" ./results\n");
      return;
    }
    Agent agent = new Agent(args[0]);
```

```
    Service service = null;
    if (args.length == 1) {
      Service[] serviceList = agent.listServices();
      for (int i=0; i<serviceList.length; i++) {
        service = serviceList[i];
        System.out.println("Service: " + service.getName()
          + service.getAbstract());
      }
      return;
    }
    service = agent.getService(args[1]);
    Job job = service.createJob();
    try {
      job.putFile(args[3]);
      job.execute(args[2]);
      ResultFile[] fileList = job.listResultFiles();
      for (int i=0; i < fileList.length; i++) {
        fileList[i].getFile(new File(args[4] + File.separator
          + fileList[i].getName()));
      }
    } finally {
      job.delete();
    }
  }
}
```

7. Условия использования. Лицензия MIT.