Java 簡易 Http クライアント

[Java]

Jakarta commons HttpClient

- http://hc.apache.org/httpclient-3.x/

SimpleHttp クライアント

上記が使えない状況があったので、HttpURLConnection を利用して簡易版を自作

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;
import java.util.List;
import java.util.Map;

import org.apache.commons.lang.StringUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.stereotype.Service;

/**
 * Web サービス利用クライアント
 */
public class ServiceClient {

    /**
     * Web サービスに、HTTP GET リクエストを送信し、結果を取得する
     * @param url
     * @param headers
     * @return
     */
    public static Response sendGetRequest(String url, Map<String, List<String>> headers) {
        return sendRequest(url, "GET", null, null, headers);
    }

    /**
     * Web サービスに、HTTP GET リクエストを送信し、結果を取得する
     * @param url
     * @param parameters
     * @param headers
     * @return
     */
     public static Response sendGetRequest(String url, Map<String, String> parameters, Map<String,
List<String>> headers) {
        return sendRequest(url, "GET", parameters, null, headers);
    }
    /**
     * Web サービスに、HTTP POST リクエストを送信し、結果を取得する
     * @param url
     * @param parameters
     * @param headers
     * @return
     */
    public static Response sendPostRequest(String url, Map<String, String> parameters , Map<String,
List<String>> headers) {
        return sendRequest(url, "POST", parameters, null, headers);
    }

    /**
     * Web サービスに、XML を伴ったリクエストを送信し、結果を取得する
     * @param url
     * @param method
     * @param xml
     * @param headers
     * @return
     */
    public static Response sendRequestWithXml(String url, HttpMethod method, String xml, Map<String,
```

```
List<String>> headers) {
        return sendRequest(url, method, null, xml , headers);
    }

    /**
     * Web サービスにリクエストを送信する
     *
     * @param url
     * @param method
     * @param parameters
     * @param xml
     * @param headers
     * @return
     */
    protected static Response sendRequest(
            String url,
            String method,
            Map<String, String> parameters,
            String xml,
            Map<String, List<String>> headers) {

        if (url == null) {
            throw new IllegalArgumentException();
        }

        boolean isGetRequst = ("GET".equalsIgnoreCase(method));
        boolean isXmlRequest = StringUtils.isNotBlank(xml);
            boolean  hasWriteTarget  =  (isXmlRequest  ||  (!isGetRequst  &&  (parameters!=null  &&
parameters.size() > 0)));
        String writeTarget = null;

        try {
            URL _url = null;

            // パラメータ文字列を生成する
            String parmString = makeParmString(parameters);

            // 出力対象
            if (hasWriteTarget) {
                if (isXmlRequest) {
                    writeTarget = xml;
                } else {
                    writeTarget = parmString;
                }
            }

            // GET の場合、クエリにパラメータ結合し URL を生成
            if (isGetRequst && StringUtils.isNotBlank(parmString)) {
                boolean hasQuery = (url.lastIndexOf('?') != -1);
                String sep = (hasQuery)?"&":"?";
                url = url + sep + parmString;
            }
            _url = new URL(url);

            HttpURLConnection conn = (HttpURLConnection) _url.openConnection();
            conn.setRequestMethod(method.toString());
            // リダイレクトを無効にする
            conn.setInstanceFollowRedirects(false);
            if (writeTarget != null) {
                // コネクションへの書き込みを有効化
                conn.setDoOutput(true);
                String size = String.valueOf(writeTarget.getBytes("UTF-8").length);
                conn.setRequestProperty("Content-Length", size);
            }

            if (headers == null) {
                // ヘッダが渡されない場合、デフォルトの値を設定
                conn.setRequestProperty("Accept-Language","ja;q=0.7,en;q=0.3");
                conn.setRequestProperty("If-Modified-Since","Thu, 01 Jun 1970 00:00:00 GMT");
                conn.setRequestProperty("Connection","Keep-Alive");
                conn.setRequestProperty("Cache-Control","no-cache");

                if (isXmlRequest) {
                    conn.setRequestProperty("Content-Type","application/xml;charset=UTF-8");
                } else if
                    ("POST".equalsIgnoreCase(method)) {
                    conn.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
                }
            } else {
                // ヘッダが渡された場合、値を転記
                for (String key: headers.keySet()) {
                    String value = StringUtils.join(headers.get(key),",");
```

```
                conn.setRequestProperty(key, value);
            }
        }

        // コネクションを開く
        conn.connect();

        // 結果の生成
        Response response = new Response();

        // レスポンスヘッダーを設定
        response.setResponseHeaders(conn.getHeaderFields());


        // 出力の書き込み
        if (writeTarget != null) {
            PrintWriter printWriter = new PrintWriter(conn.getOutputStream());
            printWriter.print(writeTarget);
            printWriter.close();
        }

        // 結果を受ける
        BufferedReader reader
            = new BufferedReader(new InputStreamReader(conn.getInputStream(), "UTF-8"));

        StringBuilder buf = new StringBuilder();
        String line = null;
        while((line = reader.readLine())!=null) {
            buf.append(line);
        }
        reader.close();

       response.setBody(buf.toString());
       return response;

    } catch (Exception e) {
       e.printStackTrace();
       throw new IllegalStateException(e);
    }
}


/**
 * パラメータ文字列を生成する
 * @param parameters
 * @return
 * @throws UnsupportedEncodingException
 */
      public  static  String  makeParmString(Map<String,  String>  parameters)  throws
UnsupportedEncodingException {
    if (parameters != null && parameters.size() > 0) {
        StringBuilder buf = new StringBuilder();

        String amp = "&";
        boolean isFirst = true;
        for (String key : parameters.keySet()){
            buf.append(((isFirst)?"":amp));
            buf.append(key + "="  + URLEncoder.encode(parameters.get(key), "utf-8") );
            isFirst = false;
        }
        return buf.toString();
    }
    return "";
}
/**
 * Web サービスクライアントのレスポンス
 */
public static class Response {
    private Map<String,List<String>> responseHeaders;
    private String body;
    public Map<String, List<String>> getResponseHeaders() {
        return responseHeaders;
    }
    public void setResponseHeaders(Map<String, List<String>> responseHeaders) {
        this.responseHeaders = responseHeaders;
    }
    public String getBody() {
        return body;
    }
    public void setBody(String body) {
        this.body = body;
    }
```

```
        /**
         * HTTP レスポンスの結果ステータスコードを取得
         * @return
         */
        public int getStatusCode() {
            try {
                List<String> statusLines = responseHeaders.get(null);
                String statusLine = statusLines.get(0);
                String[] strStatus = statusLine.split(" ");
                int status = Integer.parseInt(strStatus[1]);
                return status;
            } catch (Exception e) {
                throw new IllegalStateException(e);
            }
        }
    }
}
```