<http://blog.csdn.net/lmb55/article/details/70246635>

RestTemplate是Spring提供的用于访问Rest服务的客户端，RestTemplate提供了多种便捷访问远程Http服务的方法，能够大大提高客户端的编写效率。调用RestTemplate的默认构造函数，RestTemplate对象在底层通过使用java.net包下的实现创建HTTP 请求，可以通过使用ClientHttpRequestFactory指定不同的HTTP请求方式。

ClientHttpRequestFactory接口主要提供了两种实现方式，一种是SimpleClientHttpRequestFactory，使用J2SE提供的方式（既java.net包提供的方式）创建底层的Http请求连接，还有一种方式是使用HttpComponentsClientHttpRequestFactory方式，底层使用HttpClient访问远程的Http服务，使用HttpClient可以配置连接池和证书等信息。

**RestTemplate构造方法：**

**[java]** [view plain](http://blog.csdn.net/lmb55/article/details/70246635) [copy](http://blog.csdn.net/lmb55/article/details/70246635)

1. **public** RestTemplate(){
2. //初始化过程
3. messageConverters = **new** ArrayList();
4. errorHandler = **new** DefaultResponseErrorHandler();
5. headersExtractor = **new** HeadersExtractor();
6. messageConverters.add(**new** ByteArrayHttpMessageConverter());
7. messageConverters.add(**new** StringHttpMessageConverter());//StringHttpMessageConverter的默认编码是ISO-8859-1，可能会导致接收的返回报文出现乱码
8. messageConverters.add(**new** ResourceHttpMessageConverter());
9. messageConverters.add(**new** SourceHttpMessageConverter());
10. messageConverters.add(**new** AllEncompassingFormHttpMessageConverter());
11. **if**(romePresent)
12. {
13. messageConverters.add(**new** AtomFeedHttpMessageConverter());
14. messageConverters.add(**new** RssChannelHttpMessageConverter());
15. }
16. **if**(jaxb2Present)
17. messageConverters.add(**new** Jaxb2RootElementHttpMessageConverter());
18. **if**(jackson2Present)
19. messageConverters.add(**new** MappingJackson2HttpMessageConverter());
20. **else**
21. **if**(jacksonPresent)
22. messageConverters.add(**new** MappingJacksonHttpMessageConverter());
23. }
25. //实现超时
26. **public** RestTemplate(ClientHttpRequestFactory requestFactory){
27. **this**();
28. setRequestFactory(requestFactory);
29. }

第一个构造方法进行默认初始化，第二个构造方法可以传入ClientHttpRequestFactory参数，因为有时我们需要对请求时间进行设置，或者对超时进行处理，ClientHttpRequestFactory接口的实现类中有timeout属性，可以满足我们的需要。  
**HTTP POST请求：**

**[java]** [view plain](http://blog.csdn.net/lmb55/article/details/70246635) [copy](http://blog.csdn.net/lmb55/article/details/70246635)

1. **public** <T> T postForObject(String url, Object request, Class<T> responseType, Object... uriVariables)
2. **throws** RestClientException
3. **public** <T> T postForObject(String url, Object request, Class<T> responseType, Map<String, ?> uriVariables)
4. **throws** RestClientException
5. **public** <T> T postForObject(URI url, Object request, Class<T> responseType) **throws** RestClientException

**示例：**

**[java]** [view plain](http://blog.csdn.net/lmb55/article/details/70246635) [copy](http://blog.csdn.net/lmb55/article/details/70246635)

1. /\*\*
2. \* 使用RestTemplate发送HTTP POST请求
3. \* @param url 请求地址
4. \* @param reqStr 请求参数
5. \* @param connectionTimeout 连接超时时间
6. \* @param socketTimeout socket超时时间
7. \* @return String 返回报文
8. \*/
9. **public** String postRest(String url, String reqStr,**int** connetionTimeOut, **int** socketTimeOut) {
11. ResponseEntity<String> responseEntity0 = **null**;
12. HttpComponentsClientHttpRequestFactory requestFactory = **new** HttpComponentsClientHttpRequestFactory();
13. requestFactory.setConnectTimeout(connetionTimeOut);
14. requestFactory.setReadTimeout(socketTimeOut);
15. RestTemplate restTemplate = **new** RestTemplate(requestFactory);
16. HttpHeaders headers = **new** HttpHeaders();
17. headers.set("Content-Type", "text/plain;charset=UTF-8");//解决请求乱码问题
18. headers.set("Accept", "text/plain;charset=UTF-8");
19. headers.set("User-Agent","CLIENT(zk/1.2.1;android/1.0;640\*480;MOT-XT800)");
20. String body=**null**;
21. **try** {
22. responseEntity0 = restTemplate.postForEntity(url,**new** HttpEntity<String>(reqStr, headers), String.**class**);
23. logger.debug("响应标头：responseEntity0.getHeaders="+ responseEntity0.getHeaders());
24. body=**new** String(responseEntity0.getBody().getBytes("ISO-8859-1"), "utf-8");//解决返回报文的乱码问题
25. logger.debug("响应字符串：responseEntity0.getBody()="+ responseEntity0.getBody());
26. } **catch** (Exception e) {
27. logger.error(" errpr happened in postRest(),", e);
28. }
29. **return** body;
30. }

**HTTP GET请求：**

**[java]** [view plain](http://blog.csdn.net/lmb55/article/details/70246635) [copy](http://blog.csdn.net/lmb55/article/details/70246635)

1. //url：请求地址  responseType：返回值类型  urlVariables：请求中需要设置的参数
2. restTemplate.getForObject(String url,Class responseType,String… urlVariables);
4. //url：请求地址  responseType：返回值类型  inParams：请求中需要设置参数的Map集合
5. restTemplate.getForObject(String url,Class responseType,Map inParams);

注意：RestTemplate默认是使用org.springframework.http.converter.StringHttpMessageConverter来解析。StringHttpMessageConverter默认用ISO-8859-1来编码，所以再接收返回报文时可能会因此出现乱码的情况。

版权声明：本文为博主原创文章，无需授权即可转载，甚至无需保留以上版权声明，转载时请务必注明作者。