Code Beispiel für SSL REST Zugriff ohne Zertifikat

```java
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.ws.rs.client.Client;
import javax.ws.rs.client.ClientBuilder;
import javax.ws.rs.client.Entity;
import javax.ws.rs.client.WebTarget;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.apache.commons.codec.digest.DigestUtils;
import org.glassfish.jersey.client.ClientConfig;
import org.glassfish.jersey.client.ClientProperties;
import org.glassfish.jersey.client.authentication.HttpAuthenticationFeature;
import org.glassfish.jersey.jackson.JacksonFeature;

/**
 * Created by Oliver Brausch on 2019-07-23.
 */
public class SSLExample {

   public static void main(String[] args) {
      String url = "https://sshrest.de";
      String user = "abc";
      String pass = "123";
      Object data = "{}";

      ClientConfig config = new ClientConfig();
      config.property(ClientProperties.SUPPRESS_HTTP_COMPLIANCE_VALIDATION, true);

      Client client = buildUnsecureClient(config).register(JacksonFeature.class);
      client.register(getAuthenticationFeature(user, pass));

      WebTarget target = client.target(url);

      Response response = target
            .request(MediaType.APPLICATION_JSON)
            .build("POST", Entity.entity(data, MediaType.APPLICATION_JSON))
            .invoke();

      String text = response.readEntity(String.class);

      System.out.println("Response:" + text);

   }

   private static Client buildUnsecureClient(ClientConfig config) {
      try {

         final SSLContext context = SSLContext.getInstance("TLSv1");
         final TrustManager[] trustManagerArray = {new NullX509TrustManager()};

         context.init(null, trustManagerArray, null);

         return ClientBuilder.newBuilder()
               .withConfig(config)
               .hostnameVerifier(new NullHostnameVerifier())
               .sslContext(context)
               .build();

      } catch (Exception ex) {
         return ClientBuilder.newClient(config);
      }
   }

   private static HttpAuthenticationFeature getAuthenticationFeature(String username, String password) {
      return HttpAuthenticationFeature.basic(username, DigestUtils.shaHex(password));
   }

   /**
    * Host name verifier that does not perform any checks.
    */
   private static class NullHostnameVerifier implements HostnameVerifier {
```

```java
        public boolean verify(String hostname, SSLSession session) {
            return true;
        }
    }

    /**
     * Trust manager that does not perform any checks.
     */
    private static class NullX509TrustManager implements X509TrustManager {
        public void checkClientTrusted(X509Certificate[] chain, String authType)
                throws CertificateException {
        }

        public void checkServerTrusted(X509Certificate[] chain, String authType)
                throws CertificateException {
        }

        public X509Certificate[] getAcceptedIssuers() {
            return new X509Certificate[0];
        }
    }

}
```