I was trying to run below program

|  |
| --- |
| **package** com.url;  **import** java.io.BufferedReader;  **import** java.io.InputStreamReader;  **import** java.net.URL;  **import** java.net.URLConnection;  **import** java.nio.charset.Charset;    /\*\*  \* **@author** Crunchify.com  \*  \*/    **public** **class** CrunchifyCallUrlAndGetResponse {    **public** **static** **void** main(String[] args) {  System.***out***.println("\nOutput: \n" + *callURL*("https://cdn.crunchify.com/wp-content/uploads/code/json.sample.txt"));  }    **public** **static** String callURL(String myURL) {  System.***out***.println("Requeted URL:" + myURL);  StringBuilder sb = **new** StringBuilder();  URLConnection urlConn = **null**;  InputStreamReader in = **null**;  **try** {  URL url = **new** URL(myURL);  urlConn = url.openConnection();  **if** (urlConn != **null**)  urlConn.setReadTimeout(60 \* 1000);  **if** (urlConn != **null** && urlConn.getInputStream() != **null**) {  in = **new** InputStreamReader(urlConn.getInputStream(),  Charset.*defaultCharset*());  BufferedReader bufferedReader = **new** BufferedReader(in);  **if** (bufferedReader != **null**) {  **int** cp;  **while** ((cp = bufferedReader.read()) != -1) {  sb.append((**char**) cp);  }  bufferedReader.close();  }  }  in.close();  } **catch** (Exception e) {  **throw** **new** RuntimeException("Exception while calling URL:"+ myURL, e);  }    **return** sb.toString();  }  } |

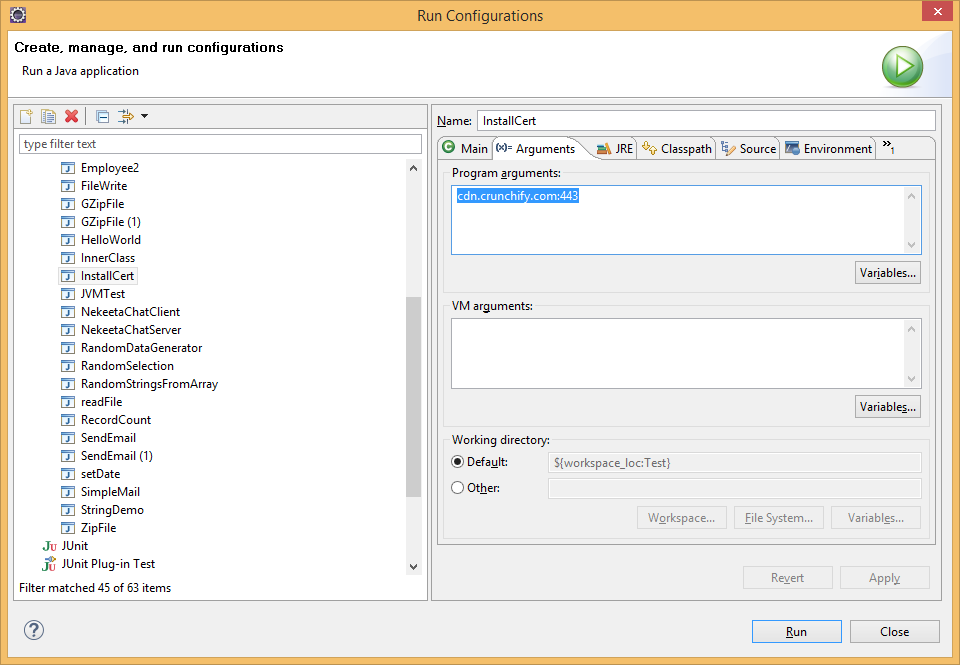
Got below issue

|  |
| --- |
| Requeted URL:https://cdn.crunchify.com/wp-content/uploads/code/json.sample.txt  Exception in thread "main" java.lang.RuntimeException: Exception while calling URL:https://cdn.crunchify.com/wp-content/uploads/code/json.sample.txt  at com.url.CrunchifyCallUrlAndGetResponse.callURL(CrunchifyCallUrlAndGetResponse.java:44)  at com.url.CrunchifyCallUrlAndGetResponse.main(CrunchifyCallUrlAndGetResponse.java:17)  Caused by: javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target  at sun.security.ssl.Alerts.getSSLException(Alerts.java:192)  at sun.security.ssl.SSLSocketImpl.fatal(SSLSocketImpl.java:1904)  at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:279)  at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:273)  at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:1446)  at sun.security.ssl.ClientHandshaker.processMessage(ClientHandshaker.java:209)  at sun.security.ssl.Handshaker.processLoop(Handshaker.java:913)  at sun.security.ssl.Handshaker.process\_record(Handshaker.java:849)  at sun.security.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:1023)  at sun.security.ssl.SSLSocketImpl.performInitialHandshake(SSLSocketImpl.java:1332)  at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:1359)  at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:1343)  at sun.net.www.protocol.https.HttpsClient.afterConnect(HttpsClient.java:559)  at sun.net.www.protocol.https.AbstractDelegateHttpsURLConnection.connect(AbstractDelegateHttpsURLConnection.java:185)  at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java:1301)  at sun.net.www.protocol.https.HttpsURLConnectionImpl.getInputStream(HttpsURLConnectionImpl.java:254)  at com.url.CrunchifyCallUrlAndGetResponse.callURL(CrunchifyCallUrlAndGetResponse.java:30)  ... 1 more  Caused by: sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target  at sun.security.validator.PKIXValidator.doBuild(PKIXValidator.java:385)  at sun.security.validator.PKIXValidator.engineValidate(PKIXValidator.java:292)  at sun.security.validator.Validator.validate(Validator.java:260)  at sun.security.ssl.X509TrustManagerImpl.validate(X509TrustManagerImpl.java:326)  at sun.security.ssl.X509TrustManagerImpl.checkTrusted(X509TrustManagerImpl.java:231)  at sun.security.ssl.X509TrustManagerImpl.checkServerTrusted(X509TrustManagerImpl.java:126)  at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:1428)  ... 13 more  Caused by: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target  at sun.security.provider.certpath.SunCertPathBuilder.engineBuild(SunCertPathBuilder.java:196)  at java.security.cert.CertPathBuilder.build(CertPathBuilder.java:268)  at sun.security.validator.PKIXValidator.doBuild(PKIXValidator.java:380)  ... 19 more |

Got below file from github <https://github.com/escline/InstallCert>

|  |
| --- |
| **package** com.url;  **import** javax.net.ssl.\*;  **import** java.io.\*;  **import** java.security.KeyStore;  **import** java.security.MessageDigest;  **import** java.security.cert.CertificateException;  **import** java.security.cert.X509Certificate;  /\*\*  \* Class used to add the server's certificate to the KeyStore  \* with your trusted certificates.  \*/  **public** **class** InstallCert {  **public** **static** **void** main(String[] args) **throws** Exception {  String host;  **int** port;  **char**[] passphrase;  **if** ((args.length == 1) || (args.length == 2)) {  String[] c = args[0].split(":");  host = c[0];  port = (c.length == 1) ? 443 : Integer.*parseInt*(c[1]);  String p = (args.length == 1) ? "changeit" : args[1];  passphrase = p.toCharArray();  } **else** {  System.***out***.println("Usage: java InstallCert <host>[:port] [passphrase]");  **return**;  }  File file = **new** File("jssecacerts");  **if** (file.isFile() == **false**) {  **char** SEP = File.***separatorChar***;  File dir = **new** File(System.*getProperty*("java.home") + SEP  + "lib" + SEP + "security");  file = **new** File(dir, "jssecacerts");  **if** (file.isFile() == **false**) {  file = **new** File(dir, "cacerts");  }  }  System.***out***.println("Loading KeyStore " + file + "...");  InputStream in = **new** FileInputStream(file);  KeyStore ks = KeyStore.*getInstance*(KeyStore.*getDefaultType*());  ks.load(in, passphrase);  in.close();  SSLContext context = SSLContext.*getInstance*("TLS");  TrustManagerFactory tmf =  TrustManagerFactory.*getInstance*(TrustManagerFactory.*getDefaultAlgorithm*());  tmf.init(ks);  X509TrustManager defaultTrustManager = (X509TrustManager) tmf.getTrustManagers()[0];  SavingTrustManager tm = **new** SavingTrustManager(defaultTrustManager);  context.init(**null**, **new** TrustManager[]{tm}, **null**);  SSLSocketFactory factory = context.getSocketFactory();  System.***out***.println("Opening connection to " + host + ":" + port + "...");  SSLSocket socket = (SSLSocket) factory.createSocket(host, port);  socket.setSoTimeout(10000);  **try** {  System.***out***.println("Starting SSL handshake...");  socket.startHandshake();  socket.close();  System.***out***.println();  System.***out***.println("No errors, certificate is already trusted");  } **catch** (SSLException e) {  System.***out***.println();  e.printStackTrace(System.***out***);  }  X509Certificate[] chain = tm.chain;  **if** (chain == **null**) {  System.***out***.println("Could not obtain server certificate chain");  **return**;  }  BufferedReader reader =  **new** BufferedReader(**new** InputStreamReader(System.***in***));  System.***out***.println();  System.***out***.println("Server sent " + chain.length + " certificate(s):");  System.***out***.println();  MessageDigest sha1 = MessageDigest.*getInstance*("SHA1");  MessageDigest md5 = MessageDigest.*getInstance*("MD5");  **for** (**int** i = 0; i < chain.length; i++) {  X509Certificate cert = chain[i];  System.***out***.println  (" " + (i + 1) + " Subject " + cert.getSubjectDN());  System.***out***.println(" Issuer " + cert.getIssuerDN());  sha1.update(cert.getEncoded());  System.***out***.println(" sha1 " + *toHexString*(sha1.digest()));  md5.update(cert.getEncoded());  System.***out***.println(" md5 " + *toHexString*(md5.digest()));  System.***out***.println();  }  System.***out***.println("Enter certificate to add to trusted keystore or 'q' to quit: [1]");  String line = reader.readLine().trim();  **int** k;  **try** {  k = (line.length() == 0) ? 0 : Integer.*parseInt*(line) - 1;  } **catch** (NumberFormatException e) {  System.***out***.println("KeyStore not changed");  **return**;  }  X509Certificate cert = chain[k];  String alias = host + "-" + (k + 1);  ks.setCertificateEntry(alias, cert);  OutputStream out = **new** FileOutputStream("jssecacerts");  ks.store(out, passphrase);  out.close();  System.***out***.println();  System.***out***.println(cert);  System.***out***.println();  System.***out***.println  ("Added certificate to keystore 'jssecacerts' using alias '"  + alias + "'");  }  **private** **static** **final** **char**[] ***HEXDIGITS*** = "0123456789abcdef".toCharArray();  **private** **static** String toHexString(**byte**[] bytes) {  StringBuilder sb = **new** StringBuilder(bytes.length \* 3);  **for** (**int** b : bytes) {  b &= 0xff;  sb.append(***HEXDIGITS***[b >> 4]);  sb.append(***HEXDIGITS***[b & 15]);  sb.append(' ');  }  **return** sb.toString();  }  **private** **static** **class** SavingTrustManager **implements** X509TrustManager {  **private** **final** X509TrustManager tm;  **private** X509Certificate[] chain;  SavingTrustManager(X509TrustManager tm) {  **this**.tm = tm;  }  **public** X509Certificate[] getAcceptedIssuers() {    /\*\*  \* This change has been done due to the following resolution advised for Java 1.7+  http://infposs.blogspot.kr/2013/06/installcert-and-java-7.html  \*\*/    **return** **new** X509Certificate[0];  //throw new UnsupportedOperationException();  }  **public** **void** checkClientTrusted(X509Certificate[] chain, String authType)  **throws** CertificateException {  **throw** **new** UnsupportedOperationException();  }  **public** **void** checkServerTrusted(X509Certificate[] chain, String authType)  **throws** CertificateException {  **this**.chain = chain;  tm.checkServerTrusted(chain, authType);  }  }  } |

Give argument cdn.crunchify.com:443 to installcert program and run it



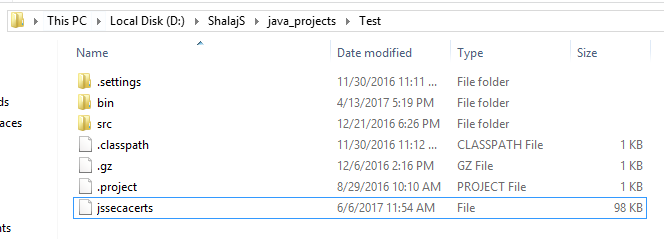
|  |
| --- |
| Loading KeyStore C:\Program Files\Java\jdk1.7.0\_80\jre\lib\security\cacerts...  Opening connection to cdn.crunchify.com:443...  Starting SSL handshake...  javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target  at sun.security.ssl.Alerts.getSSLException(Alerts.java:192)  at sun.security.ssl.SSLSocketImpl.fatal(SSLSocketImpl.java:1904)  at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:279)  at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:273)  at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:1446)  at sun.security.ssl.ClientHandshaker.processMessage(ClientHandshaker.java:209)  at sun.security.ssl.Handshaker.processLoop(Handshaker.java:913)  at sun.security.ssl.Handshaker.process\_record(Handshaker.java:849)  at sun.security.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:1023)  at sun.security.ssl.SSLSocketImpl.performInitialHandshake(SSLSocketImpl.java:1332)  at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:1359)  at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:1343)  at com.url.InstallCert.main(InstallCert.java:61)  Caused by: sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target  at sun.security.validator.PKIXValidator.doBuild(PKIXValidator.java:385)  at sun.security.validator.PKIXValidator.engineValidate(PKIXValidator.java:292)  at sun.security.validator.Validator.validate(Validator.java:260)  at sun.security.ssl.X509TrustManagerImpl.validate(X509TrustManagerImpl.java:326)  at sun.security.ssl.X509TrustManagerImpl.checkTrusted(X509TrustManagerImpl.java:231)  at sun.security.ssl.X509TrustManagerImpl.checkServerTrusted(X509TrustManagerImpl.java:107)  at com.url.InstallCert$SavingTrustManager.checkServerTrusted(InstallCert.java:163)  at sun.security.ssl.AbstractTrustManagerWrapper.checkServerTrusted(SSLContextImpl.java:885)  at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:1428)  ... 8 more  Caused by: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target  at sun.security.provider.certpath.SunCertPathBuilder.engineBuild(SunCertPathBuilder.java:196)  at java.security.cert.CertPathBuilder.build(CertPathBuilder.java:268)  at sun.security.validator.PKIXValidator.doBuild(PKIXValidator.java:380)  ... 16 more  Server sent 2 certificate(s):  1 Subject CN=cdn.crunchify.com, OU=PositiveSSL, OU=Domain Control Validated  Issuer EMAILADDRESS=sys\_admin@cybage.com, CN=Cybage Software, OU=Cybage Software Pvt. Ltd., O=Cybage Software Pvt. Ltd., L=Pune, ST=Maharashtra  sha1 df 81 fc 4c 8a 82 e8 70 83 9d bf 6f 85 06 85 57 d5 e2 03 87  md5 65 93 97 64 d1 38 c9 34 e6 1b fd 0f 4b fb 04 19  2 Subject EMAILADDRESS=sys\_admin@cybage.com, CN=Cybage Software, OU=Cybage Software Pvt. Ltd., O=Cybage Software Pvt. Ltd., L=Pune, ST=Maharashtra  Issuer EMAILADDRESS=sys\_admin@cybage.com, CN=Cybage Software, OU=Cybage Software Pvt. Ltd., O=Cybage Software Pvt. Ltd., L=Pune, ST=Maharashtra  sha1 0d ee e2 21 d9 cc fb 98 3b 96 35 98 9e 36 db c8 37 9a 63 04  md5 e9 64 df 34 dd 21 40 de 9c 77 50 c9 0b 31 e7 27  Enter certificate to add to trusted keystore or 'q' to quit: [1]  # type 1 and press enter  1  [  [  Version: V3  Subject: CN=cdn.crunchify.com, OU=PositiveSSL, OU=Domain Control Validated  Signature Algorithm: SHA256withRSA, OID = 1.2.840.113549.1.1.11  Key: Sun RSA public key, 1024 bits  modulus: 135080754032426782736058651989592298137234272285499111008754195972945010231304291591844370474640874775445310545331652218465817439802463430407811574945483140703315694792909370106612029170404745957513497783899316355601013044375953073583711492738125519769587483320083066037283487670909950564474486484982967859261  public exponent: 65537  Validity: [From: Thu Apr 06 05:30:00 IST 2017,  To: Sat Apr 07 05:29:59 IST 2018]  Issuer: EMAILADDRESS=sys\_admin@cybage.com, CN=Cybage Software, OU=Cybage Software Pvt. Ltd., O=Cybage Software Pvt. Ltd., L=Pune, ST=Maharashtra  SerialNumber: [ a0c5401a 77aa5736 d22bf133 eabb0d5c]  Certificate Extensions: 1  [1]: ObjectId: 2.5.29.17 Criticality=false  SubjectAlternativeName [  DNSName: cdn.crunchify.com  DNSName: www.cdn.crunchify.com  ]  ]  Algorithm: [SHA256withRSA]  Signature:  0000: 0C 7F F2 93 50 1A 4B 78 46 74 31 0F CC D6 1E 2E ....P.KxFt1.....  0010: AE 7F 88 C7 25 95 EA 53 3C F5 0B A1 73 55 21 7A ....%..S<...sU!z  0020: 8D 5F DC A3 69 59 9D 5D D6 FB 85 BF F8 9D 97 4B .\_..iY.].......K  0030: 69 FC D5 D4 17 81 8B 77 83 34 B2 DE 0F CB 59 CF i......w.4....Y.  0040: 25 BA A4 4C C7 8F 6B 5C 0D 2C 2F 0F 5E 32 44 5A %..L..k\.,/.^2DZ  0050: B0 10 FF 43 4A 4F AA 28 43 95 36 B3 52 8A C4 EE ...CJO.(C.6.R...  0060: 5F 39 D2 BE 46 ED B8 08 68 6E 81 6E 92 28 73 C6 \_9..F...hn.n.(s.  0070: D9 4B AF A0 EE 8F 04 4E E0 5E 67 A3 AF EF 11 7F .K.....N.^g.....  0080: B0 8A B7 9E 72 10 A0 48 EC 63 D2 96 42 4C 34 18 ....r..H.c..BL4.  0090: 11 D8 B7 BC 26 2C 98 7E E5 F7 52 E8 7B B7 C9 4E ....&,....R....N  00A0: AD 7D 05 7E 32 85 1D 49 86 FB 92 83 3A 26 C7 A2 ....2..I....:&..  00B0: E9 8B 88 2B 40 99 81 49 B0 69 B2 B5 75 F5 3B E5 ...+@..I.i..u.;.  00C0: D8 3A 18 52 AC 5F DC 80 14 4E 92 80 5F E2 66 F7 .:.R.\_...N..\_.f.  00D0: 5E 7D D3 27 85 0A 24 F0 FC 52 34 3F CE 8F 29 10 ^..'..$..R4?..).  00E0: FE 64 6B 7D D0 B0 0B CF 69 DA 27 11 AC E3 76 39 .dk.....i.'...v9  00F0: 44 CC D1 E2 41 23 14 47 87 44 E9 0B 5B DB 91 BC D...A#.G.D..[...  ]  Added certificate to keystore 'jssecacerts' using alias 'cdn.crunchify.com-1' |

To verify it again run this program, the connection should be ok now.

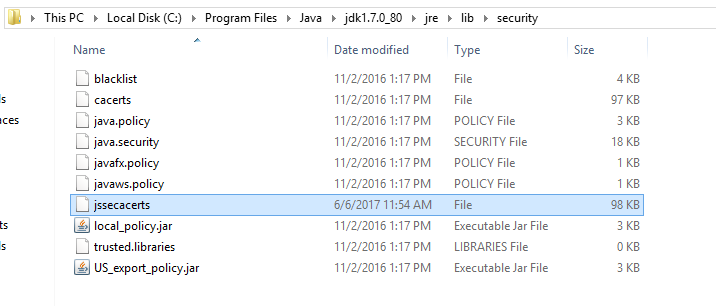
|  |
| --- |
| Loading KeyStore jssecacerts...  Opening connection to cdn.crunchify.com:443...  Starting SSL handshake...  No errors, certificate is already trusted  Server sent 2 certificate(s):  1 Subject CN=cdn.crunchify.com, OU=PositiveSSL, OU=Domain Control Validated  Issuer EMAILADDRESS=sys\_admin@cybage.com, CN=Cybage Software, OU=Cybage Software Pvt. Ltd., O=Cybage Software Pvt. Ltd., L=Pune, ST=Maharashtra  sha1 df 81 fc 4c 8a 82 e8 70 83 9d bf 6f 85 06 85 57 d5 e2 03 87  md5 65 93 97 64 d1 38 c9 34 e6 1b fd 0f 4b fb 04 19  2 Subject EMAILADDRESS=sys\_admin@cybage.com, CN=Cybage Software, OU=Cybage Software Pvt. Ltd., O=Cybage Software Pvt. Ltd., L=Pune, ST=Maharashtra  Issuer EMAILADDRESS=sys\_admin@cybage.com, CN=Cybage Software, OU=Cybage Software Pvt. Ltd., O=Cybage Software Pvt. Ltd., L=Pune, ST=Maharashtra  sha1 0d ee e2 21 d9 cc fb 98 3b 96 35 98 9e 36 db c8 37 9a 63 04  md5 e9 64 df 34 dd 21 40 de 9c 77 50 c9 0b 31 e7 27  Enter certificate to add to trusted keystore or 'q' to quit: [1]  # type q and press enter this time  q  KeyStore not changed |

Copy the generated “**jssecacerts**” file to your **“$JAVA\_HOME\jre\lib\security**” folder.

You will get jssecacers file under project folder (D:\ShalajS\java\_projects\Test)



Paste this file under <JRE>/lib/security



Now again run the first program

|  |
| --- |
| Requeted URL:https://cdn.crunchify.com/wp-content/uploads/code/json.sample.txt  Output:  {"menu": {  "id": "file",  "value": "File",  "popup": {  "menuitem": [  {"value": "New", "onclick": "CreateNewDoc()"},  {"value": "Open", "onclick": "OpenDoc()"},  {"value": "Close", "onclick": "CloseDoc()"}  ]  }  }} |

This time we get the result

Note :

If we are using some IP we need to verify host name , temporary solution is to add below code

|  |
| --- |
| **static** {  HttpsURLConnection.*setDefaultHostnameVerifier*(**new** HostnameVerifier()  {  **public** **boolean** verify(String hostname, SSLSession session)  {  // ip address of the service URL(like.23.28.244.244)  **if** (hostname.equals("52.91.35.51"))  **return** **true**;  **return** **false**;  }  });  } |

Full file

|  |
| --- |
| **package** com.url;  **import** java.io.BufferedReader;  **import** java.io.InputStreamReader;  **import** java.net.URL;  **import** java.net.URLConnection;  **import** java.nio.charset.Charset;  **import** javax.net.ssl.HostnameVerifier;  **import** javax.net.ssl.HttpsURLConnection;  **import** javax.net.ssl.SSLSession;    /\*\*  \* **@author** Crunchify.com  \*  \*/    **public** **class** AWSTomcat {  **static** {  HttpsURLConnection.*setDefaultHostnameVerifier*(**new** HostnameVerifier()  {  **public** **boolean** verify(String hostname, SSLSession session)  {  // ip address of the service URL(like.23.28.244.244)  **if** (hostname.equals("52.91.35.51"))  **return** **true**;  **return** **false**;  }  });  }    **public** **static** **void** main(String[] args) {  System.***out***.println("\nOutput: \n" + *callURL*("https://52.91.35.51:8443/"));  }    **public** **static** String callURL(String myURL) {  System.***out***.println("Requeted URL:" + myURL);  StringBuilder sb = **new** StringBuilder();  URLConnection urlConn = **null**;  InputStreamReader in = **null**;  **try** {  URL url = **new** URL(myURL);  urlConn = url.openConnection();  **if** (urlConn != **null**)  urlConn.setReadTimeout(60 \* 1000);  **if** (urlConn != **null** && urlConn.getInputStream() != **null**) {  in = **new** InputStreamReader(urlConn.getInputStream(),  Charset.*defaultCharset*());  BufferedReader bufferedReader = **new** BufferedReader(in);  **if** (bufferedReader != **null**) {  **int** cp;  **while** ((cp = bufferedReader.read()) != -1) {  sb.append((**char**) cp);  }  bufferedReader.close();  }  }  in.close();  } **catch** (Exception e) {  **throw** **new** RuntimeException("Exception while calling URL:"+ myURL, e);  }    **return** sb.toString();  }  } |