TDialog – REST-API

Innehållsförteckning

1. Inledning

Detta dokument syftar till att ge en introduktion till TDialogs REST-API.

2. Vad krävs för att REST-integrera med TDialog?

- Logisk åtkomst till TDialog.
- Lita på certifikatet som används för REST:s https-trafik
- Att TDialog litar på publika nyckeln av det nyckelpar som används för att skapa JWT:n (se auktorisation nedan).

3. Auktorisation

Authorization-headern skall finnas, vara Bearer och innehålla en giltig JSON Web Token, enligt exempel nedan. Publik nyckel till JWT-token måste läggas upp i TDialog för att TDialog ska acceptera inkommande anrop (se TDialogs konfigurationsdokumentation för detta).

JWT:n behöver innehålla följande:

- RS256-signatur
- issuer
- audience
- expire: Max 10 minuter efter skapande
- url: Den WS-URL man ska anropa. Detta gör JWT:n i praktiken one-use-only mot API:et, vilket är hela poängen.

OBS: Från TDialog 3.8 finns möjlighet att använda sessioner i REST-API:et. Om sessioner används krävs inte längre url-parametern och JWT-tokens återanvänds så länge de är giltiga och så länge serverns sessionstid inte överskrids. Se TDialogs konfigurationsdokumentation för hur sessioner konfigureras.

Nedan ett fullt exempel av en jwt-signatur:

Header:

```
{
  "alg": "RS256"
}
```

Payload:

```
{
  "aud": "td_test",
  "nbf": 1575636701,
  "iss": "testClient",
  "exp": 1575637361,
  "iat": 1575637001,
```

```
  "url": "/api/v1/user/39"
}
```

Http-request:

```
GET /api/v1/message/4711 HTTP/1.1
Host: rest.trusteddialog.se
Authorization: Bearer
eyJhbGciOiJSUzI1NiJ9.eyJhdWQiOiJ0ZF90ZXN0IiwibmJmIjoxNTc1NjM2NzAxLCJpc3Mi
OiJ0ZXN0Q2xpZW50IiwiZXhwIjoxNTc1NjM3MzYxLCJpYXQiOjE1NzU2MzcwMDEsInVybCI6I
i9hcGkvdjEvdXNlci8zOSJ9.nb9OUqkn4d7jtEbLPnsSk5zgD79ig-e-
wkWk04I3gNaqDkyWsUspPdNIp6cpbSmRECppz9lm6UGxynZN0jP8Q0sgD8KWjWC593r-
Xja5Tp395VuDo_GHr1EVKF0TsL6_CEgW4mDk8K3vxgmz3WM9Nh81BMbbFjLa3OIMYoxxvceTz
BMSrbvU-
fV12Hk9cfeQQvtJ38ZZLQLkCYdSJ5_CiGk8ZoAd_eFd1_B2eaq_12uxSbgWv5pXFo1MhdcSZ5
0SncUhi5E9pOXAxwomM_QVmm6907ZamkqkMG2r3mRZe8IqQs_3FMYrPHierGxfG6Sod2W38Qv
hOHyJUCFXpfqdsA

X-TD-CLIENT-ID: testClient
```

4. Tjänstekontrakt

Allmänna funktioner

PING

- GET /api/v1/ping
- Enda funktionen som inte kräver autentisering.
- Returnerar 200 om tjänsten är uppe.

Användare

CREATE

- POST /api/v1/user
- Skicka med ett användarobjekt som POST-data, exempelvis: {"userIdentifier":"1597653395879","emailAddress":"1597653395879@grevlinge.se","name":"Rest test user","type":"FULL",”userId”:0}
- Returvärden: 201 om användare skapats, 403 om inte behörighet, 500 om fel vid skapandet.

Se fullständigt exempel nedan.

OBS: userId:0 måste tyvärr sättas i payloaden i nuläget. Det påverkar inte funktionen alls, userId sätts när användaren skapas.

READ (id)

- GET /api/v1/user/{user-id}

- Returnerar User-objekt med det ID:t.
- Returvärden: 200 om användare hittas, 403 om inte behörighet, 500 om fel

READ (e-postadress)

- GET /api/v1/user/email/{email-address}
- E-postadressen måste vara URL- och base64-encodad
- Returnerar User-objekt med den e-postadressen.
- Returvärden: 200 om användare hittas, 404 om användare inte hittas, 403 om inte behörighet, 500 om fel

READ (identifier)

- GET /api/v1/user/identifier/{user-identifier}
- Identifieraren måste vara URL- och base64-encodad
- Returnerar User-objekt med den identifieraren.
- Returvärden: 200 om användare hittas, 404 om användare inte hittas, 403 om inte behörighet, 500 om fel

UPDATE

- PUT /api/v1/user/{user-id}
- Uppdaterar User-objektet med ID:t och värden för User objekt i request body, exempelvis:
  `{"userIdentifier":"1597653395879","emailAddress":"1597653395879@grevlinge.se","name":"Rest test user name change","userId":489,"type":"FULL"}`
- Returnerar uppdaterade User-objektet
- Returvärden: 200 om OK, 403 om inte behörighet, 500 vid fel.

OBS: Endast name och emailAddress kan ändras i TDialog. userIdentifier och userId kommer ge felmeddelande om man försöker ändra.

DELETE

- DELETE /api/v1/user/{user-id}
- Raderar User-objektet med ID
- Returvärden: 200 om OK, 403 om inte behörighet och 500 om fel.

Meddelandelåda (inkorg eller utkorg)

Inkorgar och utkorgar livscykelhanteras tillsammans med sina respektiva användare, därav att endast READ finns för meddelandekorgar.

Målet är att samtliga mottagartyper ska kunna skickas med API:et, i nuläget stöds TDialog-meddelanden, engångsmeddelanden och mina meddelanden och i 3.14 kommer SDK-meddelanden.

READ inkorg

- GET /api/v1/inbox/{user-id}/page/{page-id}
- Returnerar MessageInfo-objekt för User-objektet med UserId. Returnerar 20 objekt i taget och börjar med page-id=0.
- Returvärden: 200 om OK, 403 om inte behorighet och 500 om något går fel.

READ inkorg, antal olästa meddelanden

- GET /api/v1/inbox/{user-id}/countunread
- Returnerar antalet olästa meddelanden i användarens inbox.
- Returvärden: 200 om OK, 403 om inte behorighet och 500 om något går fel.

READ utkorg

- GET /api/v1/outbox/{user-id}/page/{page-id}
- Returnerar MessageInfo-objekt för User-objektet med UserId. Returnerar 20 objekt i taget och börjar med page-id=0.
- Returvärden: 200 om OK, 403 om inte behorighet och 500 om något går fel.

READ utkorg, antal olästa meddelanden

- GET /api/v1/outbox/{user-id}/countunread
- Returnerar antalet olästa meddelanden i användarens outbox.
- Returvärden: 200 om OK, 403 om inte behorighet och 500 om något går fel.

Meddelande

CREATE

- POST /api/v1/user/{user-id}/message
- Skapa ett draft-meddelande. Observera att det även krävs en UPDATE för att meddelandet faktiskt ska skickas.
- Returnerar meddelande-ID.
- Returvärden: 201 om skapat, 403 om inte behörighet, 500 om något gick fel

Se fullständigt exempel nedan.

READ

- GET /api/v1/message/{message-id}
- Returnerar Message-objekt med angivna ID:t
- Returvärde: 200 om OK, 403 om inte behörighet, 500 om något gick fel.

OBS: Även om READ är en GET så kommer attributet messageRead på meddelandet har uppdaterats internt av TDialog. Detta är i överenstämmelse med hur TDialog fungerar, men det gör tyvärr denna GET till en unsafe metod. För en unsafe metod, använd silent nedan, men det finns i nuläget ingen metod för att uppdatera messageRead utan att använda denna metod.

READ (utan att messageRead uppdateras)

- GET /api/v1/message/{message-id}/silent

- Returnerar Message-objekt med angivna ID:t
- Returvärde: 200 om OK, 403 om inte behörighet, 500 om något gick fel.

UPDATE (skickar meddelandet)

- PUT /api/v1/message/{message-id}
- Skickar meddelandet med angivet ID
- Headers:
  TD-SEND-NOTIFICATION: Ska e-postnotifiering skickas? (true|false)
  TD-SEND-OTP: Ska inbjudningslänk skickas? (true|false)
- I request body kan uppdateringar av meddelandet anges.
- Returvärden: 201 om meddelandet skickades, 200 om inga felmeddelanden men meddelandet ändå inte skickas (exempelvis för att meddelandet redan skickats), 403 om inte behörighet, 500 om något gick fel, 501 om mottagartypen inte stöds.

OBS 1: Endast meddelanden som är i draft-läge kan uppdateras – ett skickat meddelande kan aldrig uppdateras eller skickas igen.

OBS 2: Endast de fält (rubrik, text etc) som inte sattes när meddelandets draft gjordes kan ändras vid uppdatering.

OBS 3: För att skicka onetime-meddelanden måste extraVerification-fältet i IEnvelope fyllas, med e-postadress respektive mobiltelefonnummer till mottagande användare. Se Javadoc för IEnvelope, liksom Java-exempelkod.

DELETE

- DELETE /api/v1/message/{message-id}
- Radera meddelandet med angivet ID
- Returerar ingenting
- Returvärde: 200 om OK, 403 om inte behörighet, 500 om något gick fel.

Attachment

Ett attachment kan inte uppdateras när det väl laddats upp och raderas genom att hela meddelandet raderats, därav endast CREATE och READ.

CREATE

- POST /api/v1/attachment/message/{message-id}
- Laddar upp ett attachment. I request bodyn skall finnas base64-encodat fildata.
- Returerar ingenting
- Returvärden: 201 om OK, 200 om inget felmeddelande men filen inte lagts till (exempelvis för att en fil med samma namn redan finns), 403 om ingen behörighet, 422, om filen är för stor (jämfört med spring.http.multipart.max-file-size i application-prod.properties), 500 om något gick fel.

Se utförligt exempel nedan.

READ (lista av attachments)

- GET /api/v1/attachment/message/{message-id}
- Returnerar en lista med Attachment-objekt för angivet meddelande-id.
- Returvärden: 200 om OK, 403 om inte behörighet, 500 om något gick fel.

READ (ett attachment)

- GET /api/v1/attachment/{attachment-id}
- Returnerar Attachment-objekt med det givna ID:t
- Returvärden: 200 om OK, 403 om inte behörighet, 500 om något gick fel.

READ (attachment-filen)

- GET /api/v1/attachment/{attachment-id}/file
- Returnerar fildata för attachment-filen (base64-encodat)
- Returvärden: 200 om OK, 403 om inte behörighet, 500 om något gick fel.

5. Systemobjekt

User

Objekt som representerar en användare i TDialog. Attribut är:

- userId: Internt användar-ID. Används i webbtjänsterna
- userIdentifier: principalName, personnummer eller liknande extern identifiering av användaren.
- emailAddress: Användarens e-postadress
- name: Användarens namn
- type: Användarens typ

Se JSON-exempel nedan.

UserType

Användarens typ. Följande typer finns:

- GUEST: Gästanvändare. Oftast en medborgare
- FULL: Handläggare inom kundens organisation
- TRUSTED: Användare inom en organisation utanför kundens organisation
- UNKNOWN: Vi vet ännu inte vilken typ av användare det är. Användaren har ännu inte loggat in och tagit del av sin information.

Message

Objekt som representerar ett meddelande i TDialog. Avsändarens User har en kopia av Message-objektet i sin utkorg och alla mottagande User-objekt har en kopia vardera i sin inkorg.

Attribut är:

- messageId: Internt ID på meddelande
- fromId: ID på avsändande User-objekt
- messageTitle: Meddelanderubrik
- messageText: Meddelandetext
- sentDate: Skickatdatum (och tid)
- messageRead: Har meddelandet lästs? Detta attribut går inte att sätta explicit, utan sätts automatiskt när ett meddelande lästs, antingen genom gränssnittet eller med Web Service.
- ownerId: Vem äger meddelandet? I praktiken, i vems inkorg eller utkorg finns meddelandet?
- recipientEmails: Mottagarnas e-postadresser

Se JSON-exempel nedan.

MessageInfo

Lättviktsversion av ett meddelande, med avsikten att kunna visas i listningar. Detta är inte ett eget objekt, utan ID:t är ett meddelande-ID och samma som för motsvarande meddelande.

Attribut är:

- messageId: Internt ID på meddelande. Samma som för motsvarande ”fulla” meddelande.
- fromEmailAddress: Avsändarens e-postadress
- toEmailAddresses: Semikolonseparerad sträng med lista av e-postadresser. På avsändarmeddelandet innehåller denna samtliga mottagare, hos en mottagare innehåller den bara mottagaren själv. En mottagare kan inte se vilka andra meddelandet skickats till.
- messageTitle: Meddelanderubrik
- sentDate: Datum meddelandet skickats.

Attachment

Metadata till bifogad fil.

Attribut:

- id: Objektets ID
- fileName: Filnamn på bifogade filen.

Se JSON-exempel nedan.

6. JSON-exempel: Skapa användare, ladda upp bilagor, skicka meddelande

Förutom att känna till metoderna och systemobjekten som finns är det användbart att känna till hur TDialogs meddelandeflöde ser ut.

Meddelandeflödet är i grunden byggt för hur användargränssnittet hanterar meddelanden. När användaren skapar ett nytt meddelande (eller svarar, eller vidarebefordrar) skapas ett

meddelandeutkast i TDialog. När användaren sedan laddar upp bifogade filer läggs de till utkastet och när användaren trycker på skicka uppdateras utkastet, utkastet är inte längre ett utkast och meddelandet ”skickas”, dvs en kopia på meddelandet lägga till inkorgen hos samtliga mottagare.

I webservicegränssnittet motsvaras detta flöde av:

Create Message: Skapar ett utkast, returnerar utkastmeddelandets ID

Create Attachment (godtyckligt antal gånger): Lägger till bilagor i meddelandet.

Update Message: Skickar meddelandet.

Nedan visas ett fullständigt exempel av detta flöde:

Create User

```
POST https://rest.trusteddialog.se/api/v1/user HTTP/1.1

Accept: application/json, application/*+json
Authorization: Bearer
eyJhbGciOiJSUzI1NiJ9.eyJhdWQiOiJ0ZF90ZXN0IiwibmJmIjoxNTk3NjUzMzM1L
CJpc3MiOiJ0ZXN0Q2xpZW50IiwiZXhwIjoxNTk4NDMzMzk1LCJpYXQiOjE1OTc2NTM
zOTUsInVybCI6bnVsbH0.BYLdRMKqradOhPZVSWqyyVhABTdzxVa3zd3329O0yi8V3
JPSm2mOPOJOoAtuiLf6bMFTRI2i7sML6c6XebZV6EW_UHDX-
IJOKRY8tW_rp6pm4npTQQLUBLsZmoHVwI99c7OEST6YCqNAtCZ1v8TO0USEcURJ3pi
4qH725a2dTFFAfd2NFlxKosZSBdKMUAxI1mXd0jocQvlMQIO9YdFLMzn49DHOgGpoI
0gFHR7z29uwf31KNiNJrG4D14T97Z8K_JmJR2loYuzkXlEgSULi0uWg4ip2QZfWQed
6MdcsFB8zYCd28V2H0QZ_V5COs7I6A-UgFRWBWTQTkhhPOUCR6A
X-TD-CLIENT-ID: testClient
Content-Type: application/json
Content-Length: 127
Host: rest.trusteddialog.se
Connection: Keep-Alive
Accept-Encoding: gzip,deflate

{"userIdentifier":"1597653385879","emailAddress":"1597653385879@gr
evlinge.se","name":"Rest test user","userId":0,"type":"FULL"}
```

Användare med ID 489 returneras i vårt exempel, vilket används nedan.

Create Message

```
POST https://rest.trusteddialog.se/api/v1/user/489/message
HTTP/1.1

Accept: application/json, application/*+json
Authorization: Bearer
eyJhbGciOiJSUzI1NiJ9.eyJhdWQiOiJ0ZF90ZXN0IiwibmJmIjoxNTk3NjUzMzM1L
```

```
CJpc3MiOiJ0ZXN0Q2xpZW50IiwiZXhwIjoxNTk4NDMzMzk1LCJpYXQiOjE1OTc2NTM
zOTUsInVybCI6bnVsbH0.BYLdRMKqradOhPZVSWqyyVhABTdzxVa3zd3329O0yi8V3
JPSm2mOPOJOoAtuiLf6bMFTRI2i7sML6c6XebZV6EW_UHDX-
IJOKRY8tW_rp6pm4npTQQLUBLsZmoHVwI99c7OEST6YCqNAtCZ1v8TO0USEcURJ3pi
4qH725a2dTFFAfd2NFlxKosZSBdKMUAxI1mXd0jocQvlMQIO9YdFLMzn49DHOgGpoI
0gFHR7z29uwf31KNiNJrG4D14T97Z8K_JmJR2loYuzkXlEgSULi0uWg4ip2QZfWQed
6MdcsFB8zYCd28V2H0QZ_V5COs7I6A-UgFRWBWTQTkhhPOUCR6A
X-TD-CLIENT-ID: testClient
Content-Type: application/json
Content-Length: 82
Host: rest.trusteddialog.se
Connection: Keep-Alive
Accept-Encoding: gzip,deflate

{"message":null,"extraVerificationMap":null,"serviceAddressMap":nu
ll,"asUser":489}
```

I exemplet returneras ID 1614, dvs ID:t på det skapade meddelandeutkastet. Detta används sedan för att lägga till bilagor och skicka meddelandet.

Create Attachment

```
POST https://rest.trusteddialog.se/api/v1/attachment/message/1614
HTTP/1.1
Accept: application/json, application/*+json
Authorization: Bearer
eyJhbGciOiJSUzI1NiJ9.eyJhdWQiOiJ0ZF90ZXN0IiwibmJmIjoxNTk3NjUzMzM1L
CJpc3MiOiJ0ZXN0Q2xpZW50IiwiZXhwIjoxNTk4NDMzMzk1LCJpYXQiOjE1OTc2NTM
zOTUsInVybCI6bnVsbH0.BYLdRMKqradOhPZVSWqyyVhABTdzxVa3zd3329O0yi8V3
JPSm2mOPOJOoAtuiLf6bMFTRI2i7sML6c6XebZV6EW_UHDX-
IJOKRY8tW_rp6pm4npTQQLUBLsZmoHVwI99c7OEST6YCqNAtCZ1v8TO0USEcURJ3pi
4qH725a2dTFFAfd2NFlxKosZSBdKMUAxI1mXd0jocQvlMQIO9YdFLMzn49DHOgGpoI
0gFHR7z29uwf31KNiNJrG4D14T97Z8K_JmJR2loYuzkXlEgSULi0uWg4ip2QZfWQed
6MdcsFB8zYCd28V2H0QZ_V5COs7I6A-UgFRWBWTQTkhhPOUCR6A
X-TD-CLIENT-ID: testClient
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
Content-Disposition: attachment; filename="testuploadfile.txt"
Content-Length: 80
Host: rest.trusteddialog.se
Connection: Keep-Alive
Accept-Encoding: gzip,deflate

VGVzdCB1cGxvYWQgZmlsZS4gVGhpcyBmaWxlIGlzIHVwbG9hZGVkIGluIHRoZSB1cG
xvYWQgdGVzdC4=
```

Update Message (send)

```
PUT https://rest.trusteddialog.se/api/v1/message/1614 HTTP/1.1

Accept: application/json, application/*+json
Authorization: Bearer
eyJhbGciOiJSUzI1NiJ9.eyJhdWQiOiJ0ZF90ZXN0IiwibmJmIjoxNTk3NjUzMzM1L
CJpc3MiOiJ0ZXN0Q2xpZW50IiwiZXhwIjoxNTk4NDMzMzk1LCJpYXQiOjE1OTc2NTM
zOTUsInVybCI6bnVsbH0.BYLdRMKqradOhPZVSWqyyVhABTdzxVa3zd3329O0yi8V3
JPSm2mOPOJOoAtuiLf6bMFTRI2i7sML6c6XebZV6EW_UHDX-
IJOKRY8tW_rp6pm4npTQQLUBLsZmoHVwI99c7OEST6YCqNAtCZ1v8TO0USEcURJ3pi
4qH725a2dTFFAfd2NFlxKosZSBdKMUAxI1mXd0jocQvlMQIO9YdFLMzn49DHOgGpoI
0gFHR7z29uwf31KNiNJrG4D14T97Z8K_JmJR2loYuzkXlEgSULi0uWg4ip2QZfWQed
6MdcsFB8zYCd28V2H0QZ_V5COs7I6A-UgFRWBWTQTkhhPOUCR6A
X-TD-CLIENT-ID: testClient
Content-Type: application/json
TD-SEND-NOTIFICATION: false
TD-SEND-OTP: false
Content-Length: 370
Host: rest.trusteddialog.se
Connection: Keep-Alive
Accept-Encoding: gzip,deflate
{"message":{"messageId":1614,"fromId":489,"messageTitle":"Title,
from REST","messageText":"Text, from
REST","sentDate":"","messageRead":false,"reply":false,"recipientTy
pe":0,"inReplyToRecipientType":0,"conversationId":null,"externalOr
g":null,"ownerId":0,"recipientEmails":["1597653396427@grevlinge.se
"]},"extraVerificationMap":null,"serviceAddressMap":null,"asUser":
489}
```

Update Message (send), Mina meddelanden

Denna är identisk med Update Message (send), endast recipientType är ändrad (0 för vanliga TDialog-meddelanden, 1 för Mina meddelanden). TDialog sköter slagning mot Skatteverkets FAR-register för att skicka till korrekt brevlåda.

```
PUT /api/v1/message/60642 HTTP/1.1
Accept: application/json, application/*+json
Authorization: Bearer
eyJhbGciOiJSUzI1NiJ9.eyJhdWQiOiJ0ZF90ZXN0IiwibmJmIjoxNjE5NTE4ODk2L
CJpc3MiOiJ0ZXN0Q2xpZW50IiwiZXhwIjoxNjE5NTE5Nzk2LCJpYXQiOjE2MTk1MTg
5NTYsInVybCI6bnVsbH0.f8rkikyi34GixaZxeBAOWJu8lGXlwX8Z5rGGNE5wJKxek
M3upkrC66U7SiFas2TpshXRZeyc-
Qe8u2oFKHlOZjnOwKSRfML0yER5wo9OZOIu7ORNF_qNV2boEgCkpvl4dk_dPOd5MAR
SiX_JXlwaRO2tiTcHwOPySKPYQlLwxDeZAGFxOliQdeew9MqE2oVMihEeixpyXDi3D
mpUpbESMIA2Xnxu5kCKutEKoQwuPePTzFqeXspKTry-
```

```
BSigLVkmF5SVkdNjmWQSi3JfrNY71epqm1oYWeRXW1coRLil5Ufp-qB-
4EZVHYqzhdtxh0RaOpWEJZ86PAzI0tURCQORLg
X-TD-CLIENT-ID: testClient
Content-Type: application/json
TD-SEND-NOTIFICATION: false
TD-SEND-OTP: false
Content-Length: 391
Host: test.trusteddialog.se
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.5.5 (Java/1.8.0_191)
Cookie: JSESSIONID=F5DBF42DE78B813E02651BF82C9D7A2A
Accept-Encoding: gzip,deflate
{"message":{"messageId":60642,"fromId":1259,"messageTitle":"Title,
from REST","messageText":"Text, from
REST","sentDate":"","messageRead":false,"reply":false,"recipientTy
pe":1,"inReplyToRecipientType":0,"conversationId":null,"externalOr
g":null,"ownerId":0,"recipientEmails":["197504210456"]},"extraVeri
ficationMap":null,"serviceAddressMap":null,"asUser":22}
```

7. Exempelkod (Java)

För att underlätta integration har TDialog utvecklat en Java exempelkod som konsumerar web servicarna. Denna finns tillgänglig på tdialog.com och dess syfte är att visa anrop och användning av API:et.

Att installera exempelkoden

Koden kommer konfigurerad för anrop mot en testserver, för att förkorta steget till en fungerande REST-anslutning. Sedan kan man stegvis anpassa den till sin egen TDialog och sin egen användning.

Gör så här för att installera.

- Installera Java SDK 1.8.
- Sätt en miljövariabel JAVA_HOME som pekar på Java SDK-katalogen.
- Installera Maven https://maven.apache.org/install.html
- Ladda ner och packa upp projektet: https://www.tdialog.com/assets/other/resttest.zip
- Öppna en kommandoprompt, i Windows en CMD (inte PowerShell).
- Gå till mappen resttest
- Kör följande kommando (på en rad):
  ```
  mvn install:install-file -Dfile=lib/integration.jar
  -DgroupId=com.trusteddialog -DartifactId=integration
  -Dversion=3.10.0 -Dpackaging=jar
  ```
- Kör följande kommando:
  ```
  mvn spring-boot:run
  ```

- Nu ska exempelprojektet ha genomfört en testkörning, dvs skapat användare, skickat meddelanden etc. I kommandoprompten syns resultatet av testningen.

OBS: Den server (rest.trusteddialog.se) som är uppsatt som enkelt test för rest-integration är uppe enligt best effort och är inte en infrastrukturkomponent att räkna med i ett projekt. Meddela oss gärna om den skulle vara nere så tar vi upp den igen, men den är inte del av något supportåtagande. Den får endast användas tillsammans med exempelkoden, inte något annat syfte. Visar det sig inte fungera kommer vi att stänga ner den.

Att byta certifikat i exempelkoden

TDialogs REST-gränssnitt kräver ett RSA-nyckelpar för sin autentisering (se Authorization). Det finns många sätt att skapa ett sådant nyckelpar, nedan beskrivs ett flöde med hjälp av openssl.

Kör följande i openssl.

```
# Generate a 2048-bit RSA private key
$ openssl genrsa -out private_key.pem 2048

# Convert private Key to PKCS#8 format (so Java can read it)
$ openssl pkcs8 -topk8 -inform PEM -outform DER -in private_key.pem -out
private_key.der -nocrypt

# Output public key portion in DER format (so Java can read it)
$ openssl rsa -in private_key.pem -pubout -outform DER -out
public_key.der
```

Byt namn på private_key.der till private.der och ersätt filen som redan finns i src/main/resources i exempelprojektet.

På TDialog-servern, konfigurera in public_key.der. Se avsnittet om Web Services i konfigurationsdokumentationen.