13.4 EPS-Client

Introduction

EPS-Client is a client library that allows client application to simply call the services exposed by DMXML or EPS-PSL.

Preparation

1. Add latest version of eps-client-http to the project.
2. Import spring configuration (contained in eps-client-core.jar):

```
<import resource="classpath:/META-INF/eps-client.xml"/>
```

3. Create eps-client.properties somewhere in the classpath (on the servers in etc, for tests in src/test/resources) with the addresses of the EPS services (example for DEV):

```
# Url for the EPS-PSL server
epspsl.baseurl=http://www.epspsldv.ep.parl.union.eu/eps-psl/

# Url for the DM-XML server
dmxml.baseurl=http://www.dmxmldv.ep.parl.union.eu/dmxml/
```

Remark (Olaf, 09/03/2016):

- The examples below use SimpleTicket but I couldn't find this class. I used LoginCallerAppTicket instead.
- The same for Consumer, I couldn't find an implementation and created my own. At least for the convert service it is sufficient to implement accept(InputStream).

Services

It exposes the following services:

Convert service in different format

The following example is a conversion of an XML4EP file identified either by its UBI (resp /a ZIP file) descriptor into the following formats:
doc (docep),(resp /docx(docep)/pdf/epub/pdfReport/pdfRop/ConvertToLanguage).
It is also possible to convert the file Language

```
...
import eu.europa.europarl.eps.xml4ep.struct.Xml4EpIdentifier
(resp /import eu.europa.europarl.eps.zip.ZipArchiveFactory;  to convert a ZipArchive)
import eu.europarl.europa.eps.client.service.ConvertService;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;
import eu.europarl.europa.eps.client.util.Consumers;
(and import eu.europa.europarl.eps.spi.Language; to change the Xml4EP
)...
private static final Ticket TICKET_AT4AM = new SimpleTicket ("AT4AM", "AT4AM");

@Resource(name = ConvertService.BEAN_NAME)
private ConvertService _client;

...

    Xml4EpIdentifier identifier = Xml4EpIdentifier.parse
("eu.europa.europarl-DIN1-2013-8456135487_01.00-en-01.00_text-xml");
```

...for doc(docep) format ...

```
   _client.doc (TICKET_AT4AM, identifier, Consumers.toFile(new File("r:/test.doc")));
```

Resp/

```
    _client.doc(ticket, ZipArchiveFactory.open(new
File("R:/temp/repo/zip/valid_recursive.zip")),
    Consumers.toFile(new File("r:/test.doc")));
```

... for docx(docep) format...

```
   _client.docx (TICKET_AT4AM, identifier, Consumers.toFile(new File("r:/test.doc")));
```

Resp/

```
    _client.docx(ticket, ZipArchiveFactory.open(new
File("R:/temp/repo/zip/valid_recursive.zip")),
    Consumers.toFile(new File("r:/test.doc")));
```

... for pdf generic forrmat...

```
   _client.pdf (TICKET_AT4AM, identifier, Consumers.toFile(new File("r:/test.doc")));
```

Resp/

```
    _client.pdf(ticket, ZipArchiveFactory.open(new
File("R:/temp/repo/zip/valid_recursive.zip")),
    Consumers.toFile(new File("r:/test.doc")));
```

... for epub format ...

```
   _client.epub (TICKET_AT4AM, identifier, Consumers.toFile(new File("r:/test.doc")));
```

Resp/

```
    _client.epub(ticket, ZipArchiveFactory.open(new
File("R:/temp/repo/zip/valid_recursive.zip")),
    Consumers.toFile(new File("r:/test.doc")));
```

...for pdf format for a Report ...

```
   _client.pdfReport (TICKET_AT4AM, identifier, Consumers.toFile(new File("r:/test.doc")));
```

Resp/

```
    _client.pdfReport(ticket, ZipArchiveFactory.open(new
File("R:/temp/repo/zip/valid_recursive.zip")),
    Consumers.toFile(new File("r:/test.doc")));
```

...for pdf format for a the RoP ...

```
  _client.pdfRop (TICKET_AT4AM, identifier, Consumers.toFile(new File("r:/test.doc")));
```

Resp/

```
    _client.pdfRop(ticket, ZipArchiveFactory.open(new
File("R:/temp/repo/zip/valid_recursive.zip")),
    Consumers.toFile(new File("r:/test.doc")));
```

...for conversion of the XML4EP file language ...

```
 _client.convertToLanguage(TICKET_AT4AM, Language.FR ,
Xml4EpIdentifier.parse("eu.europa.europarl-IBI1-2014-0000001075_01.00-it-01.00_text-xml"),
Consumers.toByteArrayOutputStream(baos));
```

where 'identifier' is the identifier of the preface document and 'consumer' the Consumer object that treats the result.

CoverDate service

Updates the cover date and the document type of the Xml4EP document identified by its UBI:

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europarl.europa.eps.client.pojo.CoverDateUpdate;
import eu.europarl.europa.eps.client.pojo.CoverDateUpdateResponse;
import eu.europarl.europa.eps.client.pojo.Xml4EpSemanticCreate;
import eu.europarl.europa.eps.client.pojo.Xml4epSemanticCreateResponse;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;

...
private final Ticket ticket = new SimpleTicket ("TORIS", "TORIS");

...
@Resource(name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;

   ... For an Amendment...

CoverDateUpdateResponse response = _client.epsPslFacade().coverDateService().updateAmL
et, new
CoverDateUpdate("eu.europa.europarl-DIN1-2014-0000001096_01.00-xm-01.00_text-xml","2011-
"AM_Com_LegReport"));


   ... For a Report...

CoverDateUpdateResponse response = _client.epsPslFacade().coverDateService().updateRepo
et, new
CoverDateUpdate("eu.europa.europarl-DIN1-2016-0000001813_01.00-en-01.00_text-xml","2000-
"PR_INI"));
```

Fetch service

Returns a content (Xml, Xml4EP, media or archive) in function of a UBI; the different fetch services are:

Fetch XML4Ep

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;
import eu.europarl.europa.eps.client.util.Consumers;

...
private final Ticket ticket = new SimpleTicket("TORIS", "TORIS");

...
@Resource(name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;

    ...
    _client.getFetchService ().fetchXml4Ep (ticket,

"eu.europa.europarl-IBI1-2013-0000003290_01.00-en-01.00_text-xml",
                                                 Consumers.toFile(new File("r:/result.xml")));
```

Fetch Media

Will fetch the media file with the passed ubi

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;
import eu.europarl.europa.eps.client.util.Consumers;

...
private final Ticket ticket = new SimpleTicket ("TORIS", "TORIS");

...
@Resource (name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;

    ...
    _client.getFetchService ().fetchMedia (ticket,

"eu.europa.europarl-IMI1-2012-0000001000_01.00-en-01.00_image-png",
                                                    Consumers.toFile(new File("r:/toto.png")));
```

Fetch zip

The archive contains the document identified by the UBI passed as a parameter and, it they exist, the children of this document.

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europa.europarl.eps.xml4ep.struct.Xml4EpIdentifier;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;
import eu.europarl.europa.eps.client.util.Consumers;

......
private final Ticket ticket = new SimpleTicket ("TORIS", "TORIS");

...
@Resource (name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;

   ...
   Xml4EpIdentifier identifier = Xml4EpIdentifier.parse
("eu.europa.europarl-DIN1-2013-8456135487_01.00-en-00.00_text-xml");
   ZipArchive archive = _client.fetchZip (ticket, identifier);
```

Fetch latest XML

The archive contains the latest linguistic version of the document identified by the UBI passed as a parameter and, it they exist, the children of this document.

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europa.europarl.eps.xml4ep.struct.Xml4EpIdentifier;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;
import eu.europarl.europa.eps.client.util.Consumers;


......
private final Ticket ticket = new SimpleTicket ("TORIS", "TORIS");


...
@Resource (name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;
   ...
   Xml4EpIdentifier identifier = Xml4EpIdentifier.parse ("eu.europa.europarl-DIN1-2013-8456
135487_01.00-en-01.00_text-xml");
   _client.getFetchService ().fetchXml4EpLatest (ticket,

"eu.europa.europarl-IBI1-2013-0000003290_01.00-en-01.00_text-xml",
                                             Consumers.toFile(new File("r:/result.xml")));
```

Fetch latest Ubi

The LatestUbi string contains the last UBI version of this Xml4Ep file passed as identifier.

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europa.europarl.eps.xml4ep.struct.Xml4EpIdentifier;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;

......
private final Ticket ticket = new SimpleTicket ("TORIS", "TORIS");

...
@Resource (name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;
   ...
   Xml4EpIdentifier identifier = Xml4EpIdentifier.parse ("eu.europa.europarl-DIN1-2013-8456
135487_01.00-en-01.00_text-xml");
   String LatestUbi = _client.fetchXml4EpLatestUbi (ticket, identifier);
```

Fetch PWC User

The PwcUser string contains the document current Private Working Copy’s ID or an empty string if the document is not checked out.

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europa.europarl.eps.xml4ep.struct.Xml4EpIdentifier;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;

......
private final Ticket ticket = new SimpleTicket ("TORIS", "TORIS");

...
@Resource (name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;
   ...
   Xml4EpIdentifier identifier = Xml4EpIdentifier.parse ("eu.europa.europarl-DIN1-2013-8456
135487_01.00-en-01.00_text-xml");
   String PwcUser = _client.fetchPWCUser (ticket, identifier);
```

Instantiation service

Instanciation service returns to the consumer the content of the generated document; the map passed as a parameter contains all necessary data required for the instantiation.

```
...
import eu.europarl.europa.eps.client.service.InstantiateService;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;

...
@Resource(name = InstantiateService.BEAN_NAME)
private InstantiateService _init;

...
private static final Ticket TICKET_AT4AM = new SimpleTicket("AT4AM", "AT4AM");


 ... amendmendList service method ...
     _init.amendmentList(TICKET_AT4AM, Language.EN, InstantiateService.TEMPLATE_AM_COM_LEG, null, Consumers.toFile(new
File("r:/test.xml")));


... init service method ...
      Map <String, String> params = ...;
      _init.init(TICKET_AT4AM, params, consumer);

 ... reportInstantiation service method ...
     File file = new  File("R:\\instantiation.json");
     String UDI = _init.reportInstantiation(TICKET_AT4AM  , "2010/1234(GPA)" ,file);
```

Xml4EP save Service

In response you get the UBI of the document saved in PureXML depending of the .zip passed as parameter

```
import eu.europarl.europa.eps.client.EpsClientFacade;
import eu.europa.europarl.eps.xml4ep.struct.Xml4EpIdentifier;
import eu.europarl.europa.eps.client.ticket.SimpleTicket;
import eu.europarl.europa.eps.client.ticket.Ticket;

......
private final Ticket ticket = new SimpleTicket ("TORIS", "TORIS");

...
@Resource (name = EpsClientFacade.BEAN_NAME)
private EpsClientFacade _client;
    ...
    Xml4EpIdentifier identifier = Xml4EpIdentifier.parse
("eu.europa.europarl-DIN1-2013-8456135487_01.00-en-01.00_text-xml");


...saveZip method...
Xml4EpSemanticCreateResponse response =
_client.epsPslFacade().xml4EpSaveService().saveZip(ticket, new
File("r:/QPwithdocnumber.zip"));
String udi = response.getUdi();
```

Information about the versionin

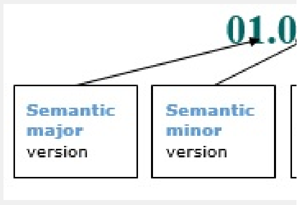
[...semanticNew method : creates a new entry]()
Xml4EpSemanticCreateRequest meta = new Xml4EpSemanticCreateRequest4Am(...);
Resp/ **meta = new Xml4EpSemanticCreateRequest4Cre(...);**
AkomaNtoso<T> akomaNtoso =...
Xml4EpSemanticCreateResponse responseExplenatoryStatement = _epsPslFacade.xml4EpSaveService().semanticNew(ticket, akomaNtoso, meta);

[...semanticMajor method : saves and updates the Major version (= if the last version is 1.24, it will create 2.0)]()
Xml4EpSemanticUpdateRequest meta = new Xml4EpSemanticUpdateRequest(...);
AkomaNtoso<T> akomaNtoso =...
Xml4EpSemanticCreateResponse responseExplenatoryStatement = _epsPslFacade.xml4EpSaveService().semanticMajor(ticket, akomaNtoso, meta);

[...semanticMinor method : saves and updates the Minor version (= if the last version is 1.24, it will create 1.25)]()
Xml4EpSemanticUpdateRequest meta = new Xml4EpSemanticUpdateRequest(...);
AkomaNtoso<T> akomaNtoso =...
Xml4EpSemanticCreateResponse responseExplenatoryStatement = _epsPslFacade.xml4EpSaveService().semanticMinor(ticket, akomaNtoso, meta);

[...linguisticNew method : creates a new entry]()
Xml4EpSemanticCreateRequest meta = new Xml4EpSemanticCreateRequest4Am(...);
Resp/ **meta = new Xml4EpSemanticCreateRequest4Cre(...);**
AkomaNtoso<T> akomaNtoso =...
Xml4EpSemanticCreateResponse responseExplenatoryStatement = _epsPslFacade.xml4EpSaveService().linguisticNew(ticket, akomaNtoso, meta);

[...linguisticMajor method : saves and updates the Major version]()
Xml4EpLinguisticUpdateRequest meta = new Xml4EpLinguisticUpdateRequest(...);
AkomaNtoso<T> akomaNtoso =...
Xml4EpSemanticCreateResponse responseExplenatoryStatement = _epsPslFacade.xml4EpSaveService().linguisticMajor(ticket, akomaNtoso, meta);



more details can be found on th

or on the "AM-Batch_PURE-XM

...linguisticMinor method : saves and updates the Minor version
Xml4EpLinguisticUpdateRequest meta = new Xml4EpLinguisticUpdateRequest(...);
AkomaNtoso<T> akomaNtoso =...
Xml4EpSemanticCreateResponse responseExplenatoryStatement = _epsPslFacade.xml4EpSaveService().linguisticMinor(ticket, akomaNtoso, meta);	

To be continued...