ADE Version 6.0

La Web Api Writer

Sommaire

Présentation générale

Concept	But des web api writer et fonctionnement général
Trames XML	Format des deux trames XML possibles pour la création de ressources ou d'activités
Réponse du service	Trame de réponse du service

1. Concept

1.1 Concept

La Web Api Writer est un web service permettant à une application d'accéder au serveur ADE afin d'y créer des ressources ou des activités.

Le principe est de fournir à l'utilisateur la possibilité d'exécuter une requête http en mode POST contenant une trame XML décrivant les ressources ou les activités à créer sous forme d'arborescence.

1.2 Exemple d'appel du service web

Il s'agit en fait d'une servlet active sur le serveur web tomcat déjà utilisé pour faire tourner ADE WEB. Contrairement à la Web API, il est nécessaire d'appeler la requête en mode POST pour y intégrer une trame XML, une simple URL paramétrer ne suffit donc pas.
La requête a appeler est : **http://ip:port/jsp/webapiwriter**

Tous les langages modernes sont capables de lancer une requête http, voici un exemple de code d'utilisation du service en utilisant java.

```
//recupération du saut de ligne
String lineSep = null;
try
{
	lineSep = System.getProperty("line.separator");
}
catch (Exception e)
{
	lineSep = "\n";
}
URL anUrl = new URL("http://ip:port/jsp/webapiwriter") ;
HttpURLConnection urlConn = (HttpURLConnection) anUrl.openConnection();
urlConn.setRequestMethod("POST");
urlConn.setAllowUserInteraction(false);

// poster des params
urlConn.setDoOutput(true);
PrintWriter paramWriter = new PrintWriter(urlConn.getOutputStream());
BufferedReader reader = new BufferedReader(new FileReader("resources.xml")) ;
while (null != (String line = reader.readLine()))
{
	paramWriter.print(line) ;
}
//fermer le post avant de lire le resultat
paramWriter.flush();
paramWriter.close();

//Lire la reponse
StringBuilder sb = new StringBuilder();
InputStream response = urlConn.getInputStream();
BufferedReader bufReader = new BufferedReader(new InputStreamReader(response));
String sLine;
while ((sLine = bufReader.readLine()) != null)
{
	sb.append(sLine);
	sb.append(lineSep);
}
//deconnection
urlConn.disconnect();
```

```
System.out.println(sb);
```

Ce code lit d'abord un fichier « resources.xml » qui contient la description des ressources à créer dans ADE, puis injecte cet XML en tant que paramètre de la requête. La servlet s'exécute au moment de la lecture de la réponse.

2. Trames XML

2.1 Connexion

Dans les 2 trames XML possibles, la connexion s'établie de la même manière, elle se fait avec le nœud «connect» dans lequel on renseigne le login et le password de l'utilisateur et l'id du projet dans lequel on va créer les ressources.

```
<connect login="root" password="" projectId="0" />
```

2.2 Ressources

Voici la DTD de la trame XML à utiliser pour la création de ressources. Cette DTD doit être présente dans la trame XML fournie au web service.

```
<!DOCTYPE resources [
<!ELEMENT resources (connect, category*)>
<!ELEMENT connect EMPTY>
<!ATTLIST connect
     login CDATA #REQUIRED
     password CDATA #REQUIRED
     projectId CDATA #REQUIRED>
<!ELEMENT category (branch*, leaf*)>
<!ATTLIST category
     category CDATA
"trainee|instructor|classroom|equipment|category5|category6|category7|category8"
     key CDATA #REQUIRED
     updates CDATA #REQUIRED>
<!ELEMENT branch (branch*, leaf*)>
<!ATTLIST branch
     key CDATA ""
     updates CDATA ""
     name CDATA ""
     code CDATA ""
     type CDATA ""
     web CDATA ""
     email CDATA ""
     quantity CDATA "-1"
     capacity CDATA "-1"
     info CDATA ""
     jobCategory CDATA ""
     manager CDATA ""
     address1 CDATA ""
     address2 CDATA ""
     zipCode CDATA ""
     city CDATA ""
     state CDATA ""
     country CDATA ""
     telephone CDATA ""
     fax CDATA ""
     codeX CDATA ""
     codeY CDATA ""
     codeZ CDATA ""
     timezone CDATA ""
     profile CDATA "all"
     owner CDATA "root"
     availabilities CDATA "Inherited">
<!ELEMENT leaf (members?, costs?, characteristics?)>
<!ATTLIST leaf
     key CDATA ""
     updates CDATA ""
```

```
      name CDATA ""
      code CDATA ""
      type CDATA ""
      web CDATA ""
      email CDATA ""
      quantity CDATA "-1"
      capacity CDATA "-1"
      info CDATA ""
      jobCategory CDATA ""
      manager CDATA ""
      address1 CDATA ""
      address2 CDATA ""
      zipCode CDATA ""
      city CDATA ""
      state CDATA ""
      country CDATA ""
      telephone CDATA ""
      fax CDATA ""
      codeX CDATA ""
      codeY CDATA ""
      codeZ CDATA ""
      timezone CDATA ""
      profile CDATA "all"
      owner CDATA "root"
      availabilities CDATA "Inherited">
<!ELEMENT members (member*)>
<!ELEMENT member EMPTY>
<!ATTLIST member
      key CDATA #REQUIRED
      keyValue CDATA #REQUIRED
      quantity CDATA "1">
<!ELEMENT costs (cost*)>
<!ELEMENT cost EMPTY>
<!ATTLIST cost
      name CDATA #REQUIRED
      value CDATA "1">
<!ELEMENT characteristics (characteristic*)>
<!ELEMENT characteristic EMPTY>
<!ATTLIST characteristic
      name CDATA #REQUIRED
      value CDATA "1">
]>
```

2.2.1 Description des ressources

On doit commencer par la catégorie :

```
<category category="trainee" key="code" updates="all,tree,availabilities">
```

Chaque description de resource se fait, soit par un noeud “branch” pour décrire un dossier, soit par un nœud leaf pour décrire une simple ressource.

```
<leaf name="A" code="GR11A" codeY="COY_GR11A" availabilities="Inherited"
profile="salle" key="code">
      <members>
            <member key="code" keyValue="GR1AB" quantity="2" />
      </members>
      <costs>
            <cost name="cout1" value="1.7" />
      </costs>
      <characteristics>
            <characteristic name="charac1" value="2.2" />
      </characteristics>
```

```
</leaf>
```

Le XML ci-dessus décrit une ressource avec son nom, son code, son codeY, son profil et sa grille de dispo (ici simplement la grille hérité). Tous les champs texte des ressources sont acceptés. Le programme ajoutera aussi la ressource dont le code est «GR1AB» avec la quantité 2, ainsi que le cout «cout1» avec la valeur 1.7 et la caractéristique «charac1» avec la valeur 2.2. Les membres, les couts et les caractéristiques ne sont ajoutés que si elles existent dans ADE.

2.2.2 Synchronisation

Pour gérer la synchronisation, on utilise les paramètres key et updates.

- key : défini la « clé primaire » de la ressource. Par exemple si on a : « key="code" », le programme va chercher une ressource dont le champ code est identique à celui de la ressource décrite. Si cette ressource existe, elle sera modifiée. Si elle n'existe pas, elle sera créée.
- updates : défini les champs à modifier en cas de modification de la ressource, séparés par une virgule. Les champs possibles sont :
  - all : tous les champs seront modifiés
  - tree : on change sa position dans l'arbre des ressources
  - availabilities : la grille de disponibilité
  - profile : le profil
  - name, type, code, … : tous les champs texte

Si ces champs ne sont pas renseignés, la valeur prise est celle de la branche supérieure, et ainsi de suite jusqu'à la catégorie s'il le faut. Key et updates doivent forcément être renseignés dans le nœud «category».

2.3 Activités

Voici la DTD de la trame XML à utiliser pour la création d'activités. Cette DTD doit être présente dans la trame XML fournie au web service.

```
<!DOCTYPE activities [
<!ELEMENT activities (connect, folder*, activity*)>
<!ELEMENT connect EMPTY>
<!ATTLIST connect
      login CDATA #REQUIRED
      password CDATA #REQUIRED
      projectId CDATA #REQUIRED>
<!ELEMENT folder (folder*, activity*)>
<!ATTLIST folder
      key CDATA ""
      updates CDATA ""
      name CDATA ""
      code CDATA ""
      priority CDATA "0"
      profile CDATA ""
      owner CDATA "">
<!ELEMENT activity (resources?) >
<!ATTLIST activity
      key CDATA ""
      updates CDATA ""
      name CDATA ""
      code CDATA ""
      type CDATA ""
      web CDATA ""
      duration CDATA ""
      repetitions CDATA ""
      info CDATA ""
      capacity CDATA ""
      timezone CDATA ""
      codeX CDATA ""
      codeY CDATA ""
      codeZ CDATA ""
      maxSeats CDATA ""
```

```
      seatsleft CDATA ""
      availabilities CDATA ""
      profile CDATA ""
      owner CDATA "">
<!ELEMENT resources (and*, or*, orRequest*)>
<!ELEMENT and EMPTY>
<!ATTLIST and
      key CDATA #REQUIRED
      keyValue CDATA #REQUIRED
      quantity CDATA "1">
<!ELEMENT or (and*, andList*)>
<!ATTLIST or
      quantity CDATA "1"
      continuous CDATA "false">
<!ELEMENT andList EMPTY>
<!ATTLIST andList
      name CDATA #REQUIRED
      quantity CDATA "1">
<!ELEMENT orRequest (and*, andList*)>
<!ATTLIST orRequest
      quantity CDATA "1"
      subject CDATA ""
      message CDATA ""
      notifyByMail CDATA "false"
      necessary CDATA "false"
      users CDATA ""
      groups CDATA ""
      sendToManager CDATA "true"
      sender CDATA "">
]>
```

2.3.1 Description des activités

Chaque description d'activité se fait dans un nœud «folder» pour les dossiers d'activité ou «activité» pour les activités elles-mêmes.

```
<activity name="Français" code="FR1" duration="4+2" repetitions="5"
codeX="COXFR1" availabilities="Standard" key="code" updates="all,tree" >
      <resources>
            <and keyValue="GR11A" key="code" quantity="1" />
            <and keyValue="AB" key="name" quantity="1" />
            <or quantity="1" continuous="true" >
                  <and keyValue="PL" key="code" quantity="1" />
                  <and keyValue="SO" key="code" quantity="1" />
            </or>
            <orRequest quantity="1" subject="sujet" message="message"
notifyByMail="false" necessary="false" users="ctirel, jdauget"
groups="Formations" sendToManager="true" sender="cvalette" >
                  <and keyValue="PL" key="code" quantity="1" />
                  <and keyValue="SO" key="code" quantity="1" />
                  <andList name="liste1" quantity="1" />
            </orRequest>
      </resources>
</activity>
```

Le XML ci-dessus décrit une activité avec son nom, son code, sa durée, ses répétitions, son code X et une grille de disponibilité standard. Il y est aussi décrit le ressources en imposés, un choisir et une requête contenant des ressources et une liste dynamique. Chaque ressource est recherchée dans ADE avec les champs «key» et «keyValue».
Par exemple, la ligne suivante :

```
<and keyValue="GR11A" key="code" quantity="1" />
```

Le programme cherchera une ressource dont le code est GR11A et l'ajoutera en imposer si elle existe.

2.3.2 Synchronisation

La synchronisation se fait de la même manière que pour les ressources. Voir le chapitre 2.2.2

3. Trame de réponse

La servlet renvoie dans son flux de réponse une trame XML qui résume le nombre d'objets créés dans ADE et les objets non trouvés :

```
<report>
     <success nbResourcesCreated="16" nbResourcesUpdated="3"
nbActivitiesCreated="0" nbActivitiesUpdated="0" nbFoldersCreated="0"
nbFoldersUpdated="0"/>
     <errors usersNotFound="" groupsNotFound="" profilesNotFound="salle"
gridsNotFound="" costsNotFound="cout1" characteristicsNotFound="charac1"
resourcesNotFound="" dynamicListsNotFound=""/>
</report>
```

Dans l'exemple ci-dessus, 16 ressources ont été créées, 3 ont été modifiés. On peut lire aussi que le profil «salle», la grille «cout1» et la caractéristique «charac1» n'étaient pas dans le projet.

Quand une erreur est générée, la trame XML est agrémentée d'un nœud correspondant à l'erreur en question. Par exemple :

```
<error type="6" name="com.adesoft.errors.EntityGroupError" />
```

Voici le tableau des erreurs susceptibles d'être rencontré :

type	Nom de l'erreur java	Cause probable
1	InvalidLogin	Le login et/ou le mot de passe est invalide
2	NotFoundException	Un objet rechercher par le programme n'a pas été trouvé, il s'agit d'un type d'erreur qui a été géré de manière particulière. Normalement, ça ne doit pas arriver
3	ProjectNotFoundException	Le numéro de projet renseigné ProjectNotFoundException
4	NoAccesException	Le user connecté n'a pas les droits ou permissions suffisantes pour les actions demandés, comme les créations de ressources ou d'activités, ou encore l'affectation de ressources à une activité.
5	SQLException	La communication entre le serveur et la base ne se fait peut-être plus.
6	EntityGroupError	On cherche à créer ou modifier une activité avec des ressources de catégories différentes dans le même choisir. Ça peut aussi se produire si la synchronisation tente de changer une ressource de place dans l'arbre, et, du coup, de modifier sa catégorie.
7	AdeException	Une autre erreur générée par ADE. Par exemple, un conflit dans les placements dus à une modification de la durée d'une activité.
-1	Autre	Erreur non prévue.