Implémentation du Web Service TagIdCheck

L'implémentation TagIdCheckRestWs de TagIdCheck correspond à un client Web Service REST envoyant des requêtes récupérant les informations de la carte par Web Service REST.

En lieu et place d'une implémentation spécifique à réaliser par vous-même, notez que vous pouvez utiliser les implémentations TagIdCheckLdap ou TagIdCheckSql permettant de récupérer ces informations de carte via une requête ou ldap ou une requête SQL.

En entrée, de ces TagIdCheck (TagIdCheckRestWs, TagIdCheckLdap ou TagIdCheckSql) le retour de badgeage tel que configuré est donné : le cas simple est l'usage du CSN mais c'est également possible de récupérer un numéro de carte (ou donnée) récupéré depusi la lecture d'un fichier via chiffrement AES au travers du protocole Desfire.

Ces configurations pourront prendre place dans la fichier applicationContext-custom.xml - par défaut un exemple y est donnée pour TagIdCheckLdap.

TagIdCheckLdap

```
<bean id="tagIdCheckApiLdapWithCsn" class="org.esupportail.nfctag.service.api.impl.TagIdCheckLdap">
 <property name="searchFilter" value="supannRefId='{'ISO15693'}'{0}"/>
 <property name="description" value="via LDAP ISO15693"/>
 <property name="ldapServices">
  <util:list>
   <ref bean="ldapServiceUnivVille"/>
  </util:list>
 </property>
</bean>
```

Si on reprend les recommandations SupAnn on pourra avoir :

```
<bean id="tagIdCheckApiLdapWithCsn" class="org.esupportail.nfctag.service.api.impl.TagIdCheckLdap">
 <property name="searchFilter" value="supannCMSId;x-mifare-xlsb={0}"/>
 <property name="description" value="via LDAP ISO15693"/>
 <property name="ldapServices">
  <util:list>
   <ref bean="ldapServiceUnivVille"/>
    <ref bean="ldapServiceUnivVille2"/>
  </util:list>
 </property>
</bean>
```

Ici on a en plus configuré l'usage de plusieurs Ldap, dans le cas où on voudrait proposer une application de badgeage multi-établissements disposant donc de plusieurs ldap.

ldapServiceUnivVille et ldapServiceUnivVille2 devant correspondre à des LdapService ainsi

```
<bean id="ldapContextSourceUnivVille"
  class="org.springframework.ldap.core.support.LdapContextSource">
 <property name="url" value="ldap://ldap.univ-ville.fr" />
 <property name="base" value="dc=univ-ville,dc=fr" />
</bean>

<bean id="ldapTemplateUnivVille" class="org.springframework.ldap.core.LdapTemplate">
 <constructor-arg ref="ldapContextSourceUnivVille" />
</bean>

<bean id="ldapServiceUnivVille" class="org.esupportail.nfctag.service.LdapService">
 <property name="ldapTemplate" ref="ldapTemplateUnivVille" />
</bean>
```

TagIdCheckSql

Pour cette implémentation, il est à noter que suivant que le badgeage consisite à lirte le CSN ou un fichier Desfire, le nom de la propriété portant la requête sql diffère.

Pour le CSN :

```
<bean id="tagIdCheckApiDonuts" class="org.esupportail.nfctag.service.api.impl.TagIdCheckSql">
 <property name="dataSource" ref="maBaseSiDataSource" />
 <property name="csnAuthSql" value="SELECT * FROM Carte WHERE csn=? AND statut=1" />
 <property name="description" value="via la base du SI"/>
</bean>
```

Pour le badgeage via un fichier d'une application Desfire :

```
<bean id="tagIdCheckApiDonuts" class="org.esupportail.nfctag.service.api.impl.TagIdCheckSql">
 <property name="dataSource" ref="maBaseSiDataSource" />
 <property name="desfireAuthSql" value="select * from EasyId where identifiantDesfireAppliEmargement=? and
statut=1" />
 <property name="description" value="via la base du SI"/>
</bean>
```

maBaseSiDataSource correspondra à un javax.sql.DataSource défini via un org.springframework.jndi.JndiObjectFactoryBean ou plus simplement via un org.apache.commons.dbcp.BasicDataSource

```
<bean class="org.apache.commons.dbcp.BasicDataSource"
               destroy-method="close" id="dataSource">
 <property name="driverClassName" value="${database.driverClassName}" />
 <property name="url" value="${database.url}" />
 <property name="username" value="${database.username}" />
 <property name="password" value="${database.password}" />
 <property name="testOnBorrow" value="true" />
 <property name="testOnReturn" value="true" />
 <property name="testWhileIdle" value="true" />
 <property name="timeBetweenEvictionRunsMillis" value="1800000" />
 <property name="numTestsPerEvictionRun" value="3" />
 <property name="minEvictableIdleTimeMillis" value="1800000" />
 <property name="validationQuery" value="SELECT version();" />
</bean>
```

TagIdCheckRestWs

TagIdCheckRestWs est utilisé lorsqu'on utilise esup-nfc-tag avec esup-sgc par exemple : esup-sgc implémente cette API TagIdCheck

Cette implémentation peut se faire dans le langage de votre choix, pour réaliser cette documentation nous nous basons ici sur une implémentation réalisée en Java (avec Spring MVC).

tagIdCheck

La fonction tagIdChek à pour but de retourver l'identifiant (login ou eppn) d'un individu en fonction de son numéro de carte (csn ou identifiant desfire).

Pour ce faire il faut implémenter une fonction ou un webservice qui accepte en entrée un csn et/ou identifiant d'un fichier d'une application desfire et qui retourne un objet « TagLog » contenant à minima :

```
@JsonIgnoreProperties(ignoreUnknown=true)
public class EsupNfcTagLog {
    String csn;
    String eppn;
    String lastname;
    String firstname;
    String location;
}
```

L'implémentation de "référence" d'eusp-sgc est donnée ici : https://github.com/EsupPortail/esup-sgc/blob/5cc2feb60725e091cde7e585bb21287a2fd076ec/src/main/java/org/esupportail/sgc/web/wsrest/WsRestEsupNfcController.java#L503C2-L529C3

```
	@RequestMapping(value="/tagIdCheck", params={"csn"}, method=RequestMethod.GET, produces = "application
/json;charset=UTF-8")
	@ResponseBody
	public EsupNfcTagLog tagIdCheck(@RequestParam String csn) {

		log.debug("tagIdCheck with csn = " + csn);

		EsupNfcTagLog esupNfcTagLog = null;
		Card card = null;

		try {
			card = Card.findCardsByCsn(csn).getSingleResult();
		} catch(Exception e){
			log.info("card not found ", e);
		}

		if(card!=null) {
			esupNfcTagLog = new EsupNfcTagLog();
			esupNfcTagLog.setCsn(card.getCsn());
			esupNfcTagLog.setEppn(card.getEppn());
			esupNfcTagLog.setFirstname(card.getUser().getFirstname());
			esupNfcTagLog.setLastname(card.getUser().getName());
			log.info("tagIdCheck OK " + esupNfcTagLog);
		} else {
			log.info("tagIdCheck failed, " + csn + " not retrieved");
		}
		return esupNfcTagLog;
	}
```