Implementation du webService AppliExtRestWs

L'implémentation AppliExtRestWs de AppliExtApi correspond à un client Web Service REST envoyant des requêtes de badgeage à un Web Service REST.

Pour intégrer le badgeage de cartes dans une de vos applications (application "métier" devant supporter le "badgeage"), vous devez donc proposer l'implémentation de ce Web Service.

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).

getLocation

La fonction getLocations retourne la liste des lieux disponibles pour l'utilisateur connecté à l'application mobile (ex : pour la carte culture le nom d'une salle de spectacle…)

La fonction prend en entrée l'identifiant de l'utilisateur (eppn) et retourne une liste de String des salles gérées par l'utilisateur.

```
@RequestMapping(value = "/getLocations", method = RequestMethod.GET, produces = "application/json;charset=UTF-
8")
@ResponseBody
public List<String> getLocations(@RequestParam String eppn) {
        List<String> listSalles = new ArrayList<String>();
        listSalles.add("Salle de test");
        return listSalles;
}
```

Exemple d'appel avec curl :

```
 curl -v https://mon-appli-metier.univ-rouen.fr/nfc-ws/getLocations?eppn=badgeur@univ-rouen.fr
```

Exemple de retour :

```
* About to connect() to mon-appli-metier.univ-rouen.fr port 443 (#0)
*   Trying xxxxxxxxx...
* Connected to mon-appli-metier.univ-rouen.fr (xxxxxxxxxx) port 443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
*   CAfile: /etc/pki/tls/certs/ca-bundle.crt
  CApath: none
* SSL connection using TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*         subject: CN=*.univ-rouen.fr,O=Université de Rouen,L=Mont-Saint-Aignan,ST=Seine-Maritime,C=FR
*         start date: mai 25 00:00:00 2016 GMT
*         expire date: mai 30 12:00:00 2019 GMT
*         common name: *.univ-rouen.fr
*         issuer: CN=TERENA SSL CA 3,O=TERENA,L=Amsterdam,ST=Noord-Holland,C=NL
> GET /nfc-ws/getLocations?eppn=badgeur@univ-rouen.fr HTTP/1.1
> User-Agent: curl/7.29.0
> Host: mon-appli-metier.univ-rouen.fr
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Wed, 24 May 2017 09:06:46 GMT
< Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
<
* Connection #0 to host mon-appli-metier.univ-rouen.fr left intact
["Inscriptions Test 1", "Inscriptions Test 2"]
```

isTagable

La fonction isTagable doit permettre de déterminer si un badge est valide pour le lieu donné. Elle prend en entrée un objet « Taglog » contenant au moins eppn et location.

La fonction retourne une entête http tel que :

HttpStatus.***OK*** (200) si l'individu est autorisé à être badgé

HttpStatus.***INTERNAL_SERVER_ERROR*** (500), accompagné d'un message dans le corps de la réponse si l'individu n'est pas autorisé

ex :

```
@RequestMapping(value = "/isTagable", method = RequestMethod.POST)
@ResponseBody
public ResponseEntity<String> eppnCheck(@RequestBody EsupNfcTagLog esupNfcTagLog) {
        if(Math.random() <= 0.5) {
                return new ResponseEntity<String>("OK", responseHeaders, HttpStatus.OK);
        } else {
                return new ResponseEntity<String>("Carte invalide", new HttpHeaders(), HttpStatus.
INTERNAL_SERVER_ERROR);
        }
}
```

Exemple d'appel avec curl :

```
curl -v -H "Content-Type: application/json" -d '{"eppn":"toto@univ-rouen.fr", "location":"Inscriptions Test
1"}' https://mon-appli-metier.univ-rouen.fr/nfc-ws/isTagable
```

Exemple de retour (erreur 500 ici : le badgeage n'est pas possible) :

```
* About to connect() to mon-appli-metier.univ-rouen.fr port 443 (#0)
*   Trying xxxxxxxxxxx..
* Connected to mon-appli-metier.univ-rouen.fr (xxxxxxxxxxx) port 443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
*   CAfile: /etc/pki/tls/certs/ca-bundle.crt
  CApath: none
* SSL connection using TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*         subject: CN=*.univ-rouen.fr,O=Université de Rouen,L=Mont-Saint-Aignan,ST=Seine-Maritime,C=FR
*         start date: mai 25 00:00:00 2016 GMT
*         expire date: mai 30 12:00:00 2019 GMT
*         common name: *.univ-rouen.fr
*         issuer: CN=TERENA SSL CA 3,O=TERENA,L=Amsterdam,ST=Noord-Holland,C=NL
> POST /nfc-ws/isTagable HTTP/1.1
> User-Agent: curl/7.29.0
> Host: mon-appli-metier.univ-rouen.fr
> Accept: */*
> Content-Type: application/json
> Content-Length: 65
>
* upload completely sent off: 65 out of 65 bytes
< HTTP/1.1 500 Erreur Interne de Servlet
< Date: Tue, 23 May 2017 10:17:57 GMT
< Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips
< Content-Type: text/plain;charset=ISO-8859-1
< Content-Length: 15
< Connection: close
<
* Closing connection 0
Carte invalide
```

validateTag

Cette fonction valide le badgeage et déclenche un traitement metier. Elle doit prendre en entrée un objet « Taglog » contenant au moins eppn et location. Elle retourne un statut http 200 si le traitement s'est bien déroulé ou une erreur http 500 dans le cas contraire.

```
@RequestMapping(value = "/validateTag", method = RequestMethod.POST)
@ResponseBody
public ResponseEntity<String> eppnValidate(@RequestBody EsupNfcTagLog esupNfcTagLog, HttpServletRequest
httpServletRequest) {
        if(Math.random() <= 0.5) {
                return new ResponseEntity<String>("OK", new HttpHeaders(), HttpStatus.OK);
        } else {
                return new ResponseEntity<String>("KO", new HttpHeaders(), HttpStatus.INTERNAL_SERVER_ERROR);
        }
}
```

Exemple d'appel avec curl :

```
 curl -v -H "Content-Type: application/json" -d '{"eppn":"toto@univ-rouen.fr", "location":"Inscriptions Test
1"}' https://mon-appli-metier.univ-rouen.fr/nfc-ws/validateTag
```

Exemple de retour (code http 200 ici : le badgeage a bien été validé) :

```
* About to connect() to mon-appli-metier.univ-rouen.fr port 443 (#0)
*   Trying xxxxxxxxxx...
* Connected to mon-appli-metier.univ-rouen.fr (xxxxxxxxxxx) port 443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
*   CAfile: /etc/pki/tls/certs/ca-bundle.crt
  CApath: none
* SSL connection using TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*         subject: CN=*.univ-rouen.fr,O=Université de Rouen,L=Mont-Saint-Aignan,ST=Seine-Maritime,C=FR
*         start date: mai 25 00:00:00 2016 GMT
*         expire date: mai 30 12:00:00 2019 GMT
*         common name: *.univ-rouen.fr
*         issuer: CN=TERENA SSL CA 3,O=TERENA,L=Amsterdam,ST=Noord-Holland,C=NL
> POST /nfc-ws/validateTag HTTP/1.1
> User-Agent: curl/7.29.0
> Host: mon-appli-metier.univ-rouen.fr
> Accept: */*
> Content-Type: application/json
> Content-Length: 65
>
* upload completely sent off: 65 out of 65 bytes
< HTTP/1.1 200 OK
< Date: Tue, 23 May 2017 10:21:02 GMT
< Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips
< Content-Type: text/plain;charset=ISO-8859-1
< Content-Length: 2
<
* Connection #0 to host mon-appli-metier.univ-rouen.fr left intact
OK
```

getDisplay

Cette fonction "POST" prend en entrée un objet TagLog et retourne un texte html. Si cette fonction est déclarée dans le AppliExtRestWs, l'application esup-nfc-(droid/desktop) affichera ce contenu textuel (html) après que l'utilisateur ait validé le badgeage (validateTag).

```
@RequestMapping(value="/display",  method=RequestMethod.POST)
@ResponseBody
public String display(@RequestBody EsupNfcTagLog taglog, Model uiModel) {
        return "<h1>Validation OK</h1><p>Ca marche !</p>";
}
```