API du validateur de format du CINES

I. Installation de l'API

voir le fichier readme.txt
Ajouter formatValidator-xxx.jar au CLASSPATH.
Les répertoires conf/ et logs/ doivent être dans le même répertoire que le JAR.

II. Interface du validateur:

Le validateur publie une interface fr.cines.format.validator.Validator

L'utilisation classique du validateur consiste en l'instanciation du validateur et l'appel des fonctions isValid() et / ou isWellFormed()

Instanciation

Pour utiliser le validateur il faut instancier la classe fr.cines.validator.Validator avec la fa brique ValidatorFactory et en indiquant un fichier à valider et d'une déclaration de format sous la forme d'une chaîne de caractères correspondant à un identifiant de format.

```
File file = new File("TEST000001.m4a");
ValidatorFactory factory = new ValidatorFactory();
Validator v = factory.createValidator(file, "AAC/AAC");
```

L'instanciation peut lever une exception `FormatValidatorException`

- si le format passé en paramètre n'est pas pris en charge
- si la version annoncée ne correspond pas à celle identifiée par le validateur

ainsi :

```
factory.createValidator(file, "GIF");
```

est moins rigoureux que

```
factory.createValidator(file, "GIF\\87a");
```

Le paramètre format est insensible à la casse ainsi GIF\\87a est identique à GIF\\87A.

Le format accepte une certaine souplesse, soit on passe l'identifiant du format (nom/enco dage\\version) tel que dans validator.xml, soit le nom du format, soit une combinaison entre nom, encodage et version.

Validation

La méthode isValid() renvoie un booléen
ex :

```
File file = new File("TEST000001.gif");
Validator v = new ValidatorFactory.createValidator(file, "GIF");
assertTrue(v.isValid());
```

La méthode isValid() appelle la méthode isWellFormed(), il est donc inutile d'appeler expli

citement les deux.

La méthode getMessage() renvoie un message explicatif sur le contrôle.

Identification

On peut utiliser l'interface Validator pour réaliser une identification
ex :

```
File f = new File("TEST000001.odt");
Validator v = new ValidatorFactory.createValidator(f, "ODT");
Format format = v.identify();
```

La méthode identify() renvoie un objet de type fr.cines.Format ou null si l'identification échoue.
L'identification est basée sur le logiciel Droid, la commande file d'unix et l'extension du fichier pour certains formats.

Attention : Si le format n'est pas présent dans validator.xml, l'identification échoue
Utilisation de la classe Format pour récupérer les informations :

String getEncoding()
List<String> getExtensionsList()

// L'identifiant au sens du validateur, format/encodage\\version. Encodage et version peuvent être nuls, à ce moment là, ils contiennent NA.
String getId()
String getLongName()
String getName()

// Liste des identifiants PRONOM associés au format
List<String> getPronomIds()
List<String> getTypeMimes()

// Version du format de fichier
String getVersion()

Ainsi pour récupérer la version d'un fichier, on peut écrire

```
Validator v = new ValidatorBean(f, "ODT");
Format format = v.identify();
format.getVersion() ;
```

Note :
Validator possède une méthode getVersion() qui renvoie la version du validateur.

III. Autre fabrique

Il est possible d'instancier la validateur sans indiquer le format comme suit :

```
File file = new File("TEST000001.m4a");
Validator v = new ValidatorFactory.createValidator(file);
```

Dans cette forme, le validateur tentera d'identifier le format en appelant sa méthode identify(). En cas d'échec il lancera une exception `FormatValidatorException.`
La validation se fera selon le format identifié.

Il est également possible d’instancier le validateur sans aucun paramètre. Ce peut être utile pour accéder à la configuration du validateur sans réaliser de validation ou pour réaliser plusieurs validations avec la même instance du validateur.