Document mis à jour le 07-04-2018 à 17h00.

[Projet BOT 2](#_Toc503554108)

[Présentation du fichier “sources.xml” : 3](#_Toc503554109)

[Présentation du fichier “words.xml” 4](#_Toc503554110)

[Présentation du fichier “output.xml” 5](#_Toc503554111)

[Algorithme du programme “bot” 6](#_Toc503554112)

# 

# Projet BOT

Le programme s’appelle bot, il est écrit en java, il se lance comme suit depuis une console shell :

|  |
| --- |
| $ java bot --sources sources.xml --words words.xml --debug false --output output.xml |

Avec :

* **bot** le programme java compilé ;
* **--sources** un paramètre obligatoire indiquant le chemin du fichier contenant la liste des sources au format XML, ce fichier sera désigné ci-après par **sources.xml** ;
* **--words** un paramètre obligatoire indiquant le chemin du fichier contenant la liste des mots au format XML, ce fichier sera désigné ci-après par **words.xml** ;
* **--output** un paramètre optionnel pour indiquer l’emplacement du fichier dans lequel stocker les articles trouvés ;
* **--debug** un paramètre optionnel pour fixer le mode de fonctionnement de l’outil.

A noter que :

* Un paramètre optionnel peut ne pas être indiqué dans la ligne de commande.
* Un paramètre obligatoire doit forcément être indiqué dans la ligne de commande.
* Le nom d’un paramètre (exemple “--sources”) est suivi obligatoirement de sa valeur (exemple “sources.xml”).
* L’ordre des paramètres ne compte pas.

Pour commencer, nous allons décrire les deux fichiers obligatoires en entrée du programme. Puis nous présenterons le format du fichier en sortie. Nous définirons ensuite la notion d’article. Nous terminerons par la description de l’algorithme du programme.

# Présentation du fichier “sources.xml” :

Exemple de contenu :

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <sources>  <source choosen=”true”>  <url>https://www.lemonde.fr</url>  <pattern><![CDATA[……..]]></pattern>  <country>FRANCE</country>  <label language=”FR-FR”>Le Monde - Actualités et Infos en France et dans le monde</label>  </source>  <source choosen=”true”>  <url>https://www.usinenouvelle.com</url>  <pattern>  <article><![CDATA[……..]]></article>  <url><![CDATA[……..]]></url>  </pattern>  <country>FRANCE</country>  <label language=”FR-FR”>L’Usine Nouvelle</label>  </source>  </sources> |

La balise racine est **sources**, cette balise contient la liste des sources d’information, chaque source d’information est contenue dans une balise **source**, qui comporte les informations suivantes :

* Un attribut, appelé **choosen**, indiquant si la source a été sélectionnée par l’utilisateur ;
* Une balise **url** contenant l’adresse url de la source d’informations ;
* Une balise **pattern** fournissant le schéma permettant de valider l’adresse Internet d’un article de presse ;
* Une balise **country** donnant le nom du pays auquel est rattaché le site d’information ;
* Une balise **label** fournissant le libellé de la source, cette balise comporte un attribut **language** fixant la langue et la région de la langue utilisée pour ce libellé.

# Présentation du fichier “words.xml”

Exemple de contenu :

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <words>  <word used=”true”>  <raw>Airbus</raw>  <clean>airbus</clean>  </word>  <word used=”true”>  <raw>eDF</raw>  <clean>edf</clean>  </word>  </words> |

La balise racine est **words**, elle contient une liste de mots, chaque mot est contenu dans une balise **word**, qui comporte les informations suivantes :

* Un attribut **used** pour indiquer si l’utilisateur souhaite ou non utiliser le mot en question ;
* Une balise **raw** comportant le mot tel que l’utilisateur l’a saisi ;
* Une balise **clean** qui correspond à la transformation du mot contenu dans la balise précédente.

# Présentation du fichier “output.xml”

Ce fichier concatène les informations que le programme produit.

Exemple de contenu :

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <articles tStart=”1515288619” tEnd=”1515288621” nArticles=”19” nNotifications=”2”>  <article>  <source>  <url>http://www.lemonde.fr/</url>  <label>Le Monde - Actualités et Infos en France et dans le monde</label>  </source>  <texte>  <raw>Entre les deux Corées, un début de détente</raw>  <clean>entre les deux corees un debut de detente</clean>  </texte>  <url>http://www.lemonde.fr/asie-pacifique/article/2018/01/09/les-deux-corees-discutent-pour-la-premiere-fois-en-deux-ans\_5239096\_3216.html</url>  <img>http://www.lemonde.fr/asie-pacifique/article/2018/01/09/les-deux-corees-discutent-pour-la-premiere-fois-en-deux-ans\_5239096\_3216.html</link>  <img>http://img.lemde.fr/2018/01/09/358/0/4000/2000/312/156/60/0/eb18cbe\_3bb6d9e665284c01b69d5ba1648fbbc4-3bb6d9e665284c01b69d5ba1648fbbc4-0.jpg</img>  <tags>  <tag>Corée</tag>  </tags>  </article>  [...]  </articles> |

La balise racine est **articles**,cette balise possède quatre attributs :

* **tStart** indique le temps de début du programme, au format POSIX ;
* **tEnd** indique le temps de fin du programme, au format POSIX ;
* **nArticles** indique le nombre d’articles ;
* **nNotifications** indique le nombre de notifications (ou le nombre d’articles dans lesquels on retrouve les mots clés de l’utilisateur).

La balise **article** contient les informations relatives à un article :

* La balise **source** contient les informations relatives à la source :
  + Son **adresse** (sous balise **url**)
  + Sa **description** (sous balise **label**)
* La balise **url** contient l’adresse complète vers le contenu de l’article ;
* La balise **img** contient l’adresse complète vers l’image associée à l’article ;
* La balise **tags** contient la liste des éventuels mots clés de l’utilisateur trouvés dans l’article, chaque mot clé est contenu dans la sous balise **tag**.

# Algorithme du programme “bot”

Séquencement des tâches effectuées par le programme :

1. **Initialisation des variables**
   1. [boolean] **debugMode** indique si le mode débuggage est activé ou pas, sa valeur est initialisé à **false**, si l’argument **--debug** est présent parmi les arguments fournis, alors on prendra la valeur suivant cet argument pour initialiser la valeur du mode de débuggage ;
   2. [boolean] **withInternet** indique si une connexion internet est possible, par défaut, sa valeur est initialisée à **false** ;
   3. [long] **tStart** instant de lancement au format POSIX;
   4. [ArrayList] **Sources** liste des sources, chaque source est un objet, au démarrage cette liste est **vide** ;
   5. [boolean] **withSources** indique si des sources sont disponibles et sélectionnées par l’utilisateur, sa valeur est initialisée à **false** ;
   6. [List] **Words** liste des mots, chaque mot est un objet, au démarrage cette liste est **vide** ;
   7. [boolean] **withWords** indique si des mots de recherche sont disponibles et activés par l’utilisateur, sa valeur est initialisée à **false** ;
   8. [List] **Articles** liste des articles, chaque article est un objet, au démarrage cette liste est **vide** ;
   9. [integer] **nArticles** indique combien d’articles ont été récupérés depuis Internet sur la base des sources à interroger, sa valeur est initialisée à **0**.
   10. [integer] **nNotifications** indique combien d’articles correspondent aux mots clés de l’utilisateur.
   11. [char(2)] **language** indique la langue de l’utilisateur, par défaut cette variable est initialisée à **en**.
   12. [string] **outputFile** indique le nom du fichier dans lequel sauvegarder les articles trouvés, sa valeur est initialisée à **output.xml**, sauf si l’argument **--output** est présent, alors on prendra la valeur attachée à cet argument.
   13. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de chaque valeur précédente avec sa valeur (exemple : “debugMode = true”).
2. **Contrôle de la présence d’une connexion Internet**
   1. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de l’étape (idem : “Step : Check Internet connection...”).
   2. Il s’agit de vérifier que le programme peut accéder à Internet, si le test est positif la variable **withInternet** est mise à **true** sinon elle est laissée à **false**.
      1. Le contrôle de la connexion se fait en interrogeant le site internet suivant : **http://www.spythenews.com/status**, si ce site est accessible alors le programme est réputé pouvoir accéder à Internet.
   3. Si **withInternet** vaut **false**, alors le programme s’arrête.
3. **Chargement des sources**
   1. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de l’étape (idem : “Step : Loading sources...”).
   2. Contrôle de la présence du fichier contenant les sources, ce fichier est passé comme argument de l’option.
   3. Si le fichier est présent, chargement des sources qu’il contient :
      1. Pour chaque source identifiée, on contrôle la présence des informations attendues, si celles-ci sont complètes, on ajoute dans **Sources** un objet ayant les propriétés suivantes :
         1. [boolean] **choosen** venant de l’attribut de la balise source ;
         2. [String] **url** venant du contenu de la balise url au sein de source ;
         3. [Pattern] **pattern** venant du contenu de la balise pattern au sein de source ;
         4. [String] **country** venant du contenu de la balise country au sein de source ;
         5. [String] **label** venant du contenu de la balise label au sein de source, par défaut, on regardera s’il existe un attribut label ayant pour attribut la langue de l’utilisateur (idem **language**), dans le cas contraire, on prendra la première balise trouvée (qui sera considérée comme la balise contenant le libellé d’origine du site d’informations) ;
      2. Ensuite, on regarde si au moins une source a été sélectionnée par l’utilisateur (**choosen** valant **true**), si tel est le cas, on passe la variable **withSources** à **true**.
   4. Si la variable **withSources** vaut **false**, alors le programme s’arrête.
4. **Chargement des mots**
   1. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de l’étape (idem : “Step : Loading words...”).
   2. Contrôle de la présence du fichier contenant les mots, ce fichiers est passé comme second argument du programme.
   3. Si le fichier est présent, chargement des mots qu’il contient :
      1. Pour chaque mot identifié, on contrôle la présence des informations attendues, si celles-ci sont complètes, on ajoute dans **Words** un objet ayant les propriétés suivantes :
         1. [boolean] **used** venant de l’attribut de la balise word ;
         2. [String] **clean** venant de la balise clean au sein de word ;
         3. [String] **raw** venant de la balise raw au sein de word ;
      2. Ensuite, on regarde si au moins un mot est utilisé (**used** valant **true**), si tel est le cas, on passe la variable **withWords** à **true**, sinon on la laisse à **false**.
   4. Si **debugMode** est égal à **true**, alors afficher dans la console le contenu de la variable **withWords** (idem : “withWords = true”).
5. **Chargement des articles**
   1. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de l’étape (idem : “Step : Loading articles...”).
   2. Prérequis, on a une connection internet (**withInternet** vaut **true**) et des sources à interroger (**withSources** vaut **true**).
   3. Pour chaque **Source** contenue dans **Sources**, si la propriété **choosen** vaut **true**, on crée un **thread** (que l’on rattachera au thread principal) qui se charge de réaliser les tâches suivantes :
      1. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de la source (exemple : “source : http://www.lemonde.fr”).
      2. On commence par récupérer le contenu de la page (dont l’adresse est stockée dans la propriété **url**), que l’on stocke dans la variable **htmlContent**.
      3. On supprime tous les sauts de ligne (à savoir **\r** et **\n**) au sein de la variable **htmlContent**.
      4. On extrait de **htmlContent** la liste des articles potentiels (confère section “définition d’un article”), les résultats sont stockés dans un tableau d’objets appelé **Results**.
      5. On libère la variable **htmlContent**.
      6. On effectue un premier nettoyage de la variable **Results** en ne conservant que les objets contenant une adresse (href=”...”) vérifiant le **pattern** de la **Source** considérée.
      7. On effectue un second nettoyage de la variable **Results** en supprimant les doublons. On a un doublon lorsque deux objets de **Results** présentent la même adresse (href=”...”) ; si tel est le cas, on ne conserve que l’objet qui porte le plus d’informations (mesuré par sa taille).
      8. On itère sur chaque objet **Result** du tableau **Results**, il vient :
         1. On initialise à vide les variables de type string suivantes : **text**, **img**, et **href**.
         2. On stocke dans **text** le contenu HTML de **Result** auquel on retire toutes les balises html par application de la substitution **s/<[^>]+//g**.
         3. On crée une variable **img** égale au contenu correspondant au motif **<img [^>]+>** cherché au sein de **text**.
         4. Si le contenu de **img** est **non vide** et que le contenu de **text** est **vide**, alors on crée une nouvelle variable **alt** contenant la sous-chaîne correspondant au motif **alt=”[^"]+”** trouvée dans **img**, si cette variable n’est pas vide, alors on stocke son contenu dans **text**, et on supprime la variable **alt**.
         5. Si le contenu de **img** est **non vide**, on remplace son contenu par l’adresse de l’image qu’elle contient, cette adresse répond au masque **src=”[^"]+”** que l’on doit appliquer au contenu de la variable **img**.
         6. Enfin, on cherche l’adresse de l’article au sein de **Result** par application du masque **<a .\*?href=”[^"]+”** au contenu de **Result**, si ce masque est trouvé, on stocke son contenu dans la variable **href**.
         7. Si **txt**, **img**, **href** sont non vides, on instancie un objet **Article** avec les propriétés suivantes :
            1. [string] **Article.sourceURL** égal à **Source.url** ;
            2. [string] **Article.sourceLabel** égal à **Source.label** ;
            3. [string] **Article.txt** égal à **txt** ;
            4. [string] **Article.clean** égal à **txt** transformé (que des minuscules ou des chiffres, aucun accent, pas de signe de ponctuation, un seul espace entre chaque mot)
            5. [string] **Article.img** égal à **img** ;
            6. [string] **Article.href** égal à **href** ;
            7. [string] **Article.keyWords** égal à un tableau vide ;
         8. On rajoute cet objet **Article** à la liste **Articles**.
         9. On incrémente la variable **nArticles**.
   4. Si **debugMode** est égal à **true**, alors afficher dans la console le contenu de la variable **nArticles** (idem : “nArticles = 19”).
6. **Identification des articles à notifier**
   1. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de l’étape (idem : “Step : Looking for notifications...”).
   2. Prérequis, on a des articles (**nArticles** **> 0**) et des mots sont à rechercher (**withWords** vaut **true**).
   3. On itère sur chaque objet **Article** du tableau **Articles**, il vient :
      1. On itère sur chaque objet **Word** du tableau **Words**, il vient :
         1. Si la séquence **Word.clean** est présente dans **Article.clean** alors on rajoute **Word.raw** au tableau **Article.keyWords**.
      2. A la fin de cette itération sur les mots clés de l’utilisateur, on regarde si le tableau **Article.keyWords** est non vide, si c’est le cas, on incrémente la variable **nNotifications**.
   4. Si **debugMode** est égal à **true**, alors afficher dans la console le contenu de la variable **nNotifications** (idem : “nNotifications = 3”).
7. **Sauvegarde des articles**
   1. Si **debugMode** est égal à **true**, alors afficher dans la console le nom de l’étape (idem : “Step : Saving articles...”).
   2. On stocke dans **tEnd** l’instant actuel au format POSIX.
   3. On stocke dans le fichier plat **outputFile** les informations trouvées précédemment, confère présentation du fichier “output.xml” faite précédemment.
   4. On arrête le programme.

oOo