

## OneSearch

***Moteur d'Indexation et de Recherche Documentaire*** ■

---

**OneCor.**  
*The e-business group*

68, Avenue Omar Ibn Khattab, Agdal - Rabat - Maroc  
Tél. : 212 (0) 37 67 41 86 - Fax : 212 (0) 37 67 42 15  
Email : [info@onecor.com](mailto:info@onecor.com)  
[www.onecor.com](http://www.onecor.com)

- Description de OneSearch
- Caractéristiques Techniques de OneSearch
- Types d'Applications Créées avec OneSearch
- Principes Fonctionnels de OneSearch
- Architecture Fonctionnelle de OneSearch

OneSearch est un outil de développement composé d'un puissant moteur multilingue d'indexation et de recherche en texte intégral.

OneSearch se compose d'une boîte à outils permettant la réalisation d'applications logicielles multi plates-formes de toute nature et de toute envergure : systèmes de gestion documentaire peu volumineux pour groupes de travail ou applications à base de serveur documentaire sur Internet gérant des millions de documents.

Constitué d'une bibliothèque de procédures et d'interfaces programmatiques, OneSearch offre une solution modulaire et flexible pour la conception ou l'intégration dans un logiciel existant de fonctionnalités d'indexation et de recherche documentaire.

Le moteur d'indexation et de recherche OneSearch est totalement indépendant de toute architecture de bases de données : le stockage des caractéristiques de chacun des documents composant le référentiel documentaire s'effectue dans un système de fichiers d'index.

OneSearch utilise le moteur de recherche en texte intégral **Lucene**, disponible en logiciel libre pour la Plate-forme de développement **Java**, ce qui rend les applications développées à partir de OneSearch :

- Multilingues
- Multi plates-formes et indépendantes de toute plate-forme système (MacOS, Linux, Windows 98, 2000, NT, Unix)
- Interfaçables avec la majorité des systèmes de Gestion de Bases de Données du marché
- Portables sur tout support de stockage (disque dur ou disque compact)
- D'une architecture totalement modulaire.

Les principales caractéristiques techniques de OneSearch sont les suivantes :

- **Moteur d'Indexation modulaire et performant :**
  - Indexation de plus de 200 Mégaoctets par heure sur machine Pentium II cadencée à 266 Mhz
  - Indexation incrémentale aussi rapide que l'indexation par lot
  - Besoins en mémoire vive peu élevés
  - Taille de l'index n'excédant pas 30% de la taille du texte indexé
- **Algorithmes de Recherche Puissants, Précis et Efficaces :**
  - Classement des résultats de la recherche (les résultats les plus pertinents sont proposés en premier)
  - Utilisation des opérateurs booléens et des phrases dans les requêtes
  - Recherche sur les champs (titre, auteur, contenu, etc.)
  - Recherche sur une plage de dates
  - Recherche sur index multiples avec fusion des résultats
  - Recherche distribuée en réseau.
- **Interfaces Programmatisées Simples :**
  - Intégration de nouveaux formats de fichiers et de documents
  - Intégration de langues supplémentaires (toutes les langues du monde sont gérées virtuellement)
  - Développement de nouvelles interfaces utilisateurs.
- **Solution totalement Multi Plates-formes:**
  - OneSearch est constitués d'une bibliothèque d'interfaces programmatique développées en **Java 100%** (certification en cours), capable de fonctionner sur toute **Machine Virtuelle Java** et compatible.

OneSearch permet le développement d'applications ou l'ajout à une application existante de fonctionnalités d'indexation et de recherche documentaire portant virtuellement sur des objets de connaissance de toute nature, liées à l'exploitation des nouveaux outils d'accès à l'information (multimédia, Internet) :

- **Serveurs documentaires et Systèmes de Gestion de Contenus :**

Une application (éventuellement intégrée au menu *Ouverture de Document*) permettant la recherche d'un contenu spécifique dans un référentiel de documents de toute nature archivés.

- **Recherche dans des Archives de Courrier Electronique :**

Une application de messagerie électronique permettant à l'utilisateur une recherche instantanée dans de gros volumes de courrier électronique archivé, avec mise à jour de l'index au fur et à mesure de l'arrivée de nouveaux messages..

- **Recherche Documentaire en Ligne :**

Un outil de navigation documentaire indépendant de toute plate-forme système offrant des fonctions de recherche rapide dans de gros volumes de documents stockés localement sur CD-ROM ou sur Internet, et intégré ou non à une application.

- **Recherche de pages Web déjà visitées :**

Un navigateur Web ou un serveur mandataire peut intégrer un moteur de recherche personnel permettant l'indexation de chaque page Web visitée, afin d'assurer la localisation rapide des pages Web visitées plusieurs semaines ou plusieurs mois auparavant.

- **Recherche dans un Site Web :**

Un outil de recherche compatible avec la norme CGI (*Common Gateway Interface*) intégré à un site Web, permettant à un internaute d'effectuer des recherches dans l'ensemble des pages du site.

- **Contrôle des Versions de Documents et Gestion de Contenu :**

Un Système de Gestion Documentaire permettant l'indexation des documents et des versions de documents afin de faciliter leur extraction.

- **Serveur de Fils d'Agences de Presse :**

Un serveur ou un relais de dépêches d'agences de presse réalisant l'indexation des articles et dépêches au fur et à mesure de leur arrivée sur le fil d'agence.

Flexible et particulièrement facile à implémenter, le logiciel **OneSearch** est une boîte à outils logicielle architecturée en **Java** offrant des fonctionnalités d'indexation et de recherche textuelle particulièrement évoluées.

Robuste et modulaire, reposant sur des principes de fonctionnement simples et puissants, le moteur d'indexation et de recherche multilingue OneSearch permet de résoudre la plupart des problématiques de recherche quel que soit le contexte professionnel et linguistique concerné.

## Le Moteur d'Indexation

Fonctionnant à partir d'un Système de Fichiers et indépendant de tout système de gestion de base de données, le moteur d'indexation de OneSearch réalise l'indexation en texte intégral de documents entiers ou l'indexation des champs définis pour ces documents.

L'index est stocké dans un répertoire spécifique constitué de fichiers. Les fichiers indexés peuvent être des fichiers texte simples ou d'autres formats de documents ou encore des documents externes (non stockés dans des fichiers).

L'indexation des fichiers se déroule en trois phases : la création de l'index, le chargement de l'index et la recherche dans l'index.

### Création de l'Index

Une fonction permet de créer un index vide en spécifiant dans la ligne de commande l'adresse du répertoire de stockage de l'index. Une option permet de réaliser le vidage du répertoire d'index en supprimant les fichiers qu'il contient.

### Ajout des Documents dans l'Index

La procédure d'indexation est initialisée par le lancement manuel d'un processus élémentaire (*thread*) effectuant l'indexation des l'ensemble des fichiers contenus dans un dossier (*folder*).

Pour chaque fichier du dossier à indexer, un objet Document est créé sous la forme d'un **fichier Alias** au format **XML**. Le fichier Alias est géré comme un ensemble de champs constitués de paires Nom du champs-Valeur du champs. Il contient deux catégories de champs qui sont soit automatiquement renseignés par l'application (les paramètres) soit saisis manuellement par l'utilisateur selon le type du document indexé lors de la procédure d'intégration du document à la bibliothèque électronique:

- Les **Paramètres d'identification** spécifiques pour chaque document, gérés par l'application :
  - **Identifiant** du document : numéro d'identification attribué automatiquement par le système (compteur incrémental).
  - **Localisation** du document : chemin d'accès, nom et extension du document.
  - **Langue du document** : l'indication de la langue conditionne le chargement du fichier de mots blancs ou mots non significatifs (*stop words*) correspondant à la langue spécifiée, ce qui optimise la durée des opérations de recherche.
  - **Action à effectuer**, c'est à dire le type d'action à réaliser dans l'index pour le document courant : ajout du document, suppression ou mise à jour.

- Les **Champs**, facultatifs et variables en fonction du type de document, se composent des éléments suivants :
  - **Nom du champs** : titre, auteur, etc.
  - **Type du champs** : texte ou numérique (chemin d'accès, nom de fichier, date, etc.)
  - **Valeur du champs** : chaîne de caractères ou chemin d'accès. La valeur des champs courts est une chaîne de caractères, celle des champs longs un flot d'entrée (*Input stream*).

OneSearch peut gérer jusqu'à 10.000 champs par document.

L'utilisation de champs permet le découpage du document en sections pouvant être indexées et recherchées séparément et l'association de méta-données au document (nom, auteur, date de modification ou de parution, etc.).

L'utilisation des champs permet également d'augmenter la portée des opérations de recherche et de construire des requêtes sémantiquement plus riches.

## Procédure d'Indexation

Le Fichier Alias est stocké dans un répertoire régulièrement scruté par un processus élémentaire (*thread*) qui effectue les opérations suivantes :

- Lecture du contenu du répertoire
- Analyse syntaxique de chaque fichier alias
- Indexation conditionnelle du document :
  - **Contrôle de l'extension du fichier** et vérification de son indexabilité (un fichier image ne sera pas indexé). OneSearch permet la création de filtres d'extension.
  - **Contrôle de la possibilité de conversion du fichier**, s'il ne s'agit pas d'une version de fichier directement exploitable (document compacté avec **WinZip** par exemple ou fichier portable au format PDF) : le moteur d'indexation crée d'une version temporaire contenant la version convertie, indexe de cette version du fichier, puis supprime le fichier temporaire.
  - **Contrôle de la langue du fichier** : chargement du dictionnaire des mots blancs (stop words) correspondants à la langue choisie. Les mots blancs sont des mots communs sans valeur sémantique qui ne sont pas indexés pour éviter d'augmenter exagérément le temps d'indexation et la taille de l'index : articles, conjonctions, etc.
  - Lecture du texte, filtrage des mots blancs (le filtre ainsi construit sera réutilisé ultérieurement pour les documents de la même langue) et marquage des mots à indexer.
- Création d'un objet document virtuel à qui sont affectées les valeurs contenues dans les paramètres et les champs renseignés.

- Stockage de chaque mot indexé et de son inverse dans une base de données créée avec **MySQL** et contenant, pour chaque langue gérée, une table de mots par lettre du jeu de caractères correspondant à cette langue :
  - Stockage du mot dans une table correspondant à la première lettre de ce mot
  - Stockage de l'inverse du mot dans une table correspondant à la dernière lettre de ce mot.

En fonction du type d'action mentionné dans le paramètre **Action**, la procédure d'indexation se déroule de la manière suivante :

- Ajout d'un document à l'index :
  - Lecture du type d'action **Ajout** dans le fichier alias
  - Ajout de l'objet document à l'index
  - Suppression du fichier alias.
- Suppression d'un document de l'index :
  - Lecture du type d'action Suppression dans le fichier alias,
  - Création d'un objet documentaire contenant uniquement les paramètres
  - Suppression du fichier alias.
- Mise à jour d'un document déjà indexé :
  - Lancement de la procédure de suppression du document
  - Lancement de la procédure d'ajout d'un document.

Une fois la procédure d'indexation terminée, les résultats s'affichent dans la fenêtre des fichiers indexés : identifiant, chemin d'accès du fichier indexé, type d'action effectuée, langue du fichier et durée de la procédure d'indexation en millièmes de secondes,

Après avoir supprimé les fichiers alias et effectué l'indexation des documents, l'agent contrôleur d'indexation se met en sommeil et scrute par intervalles les demandes d'indexation du système.

Une fonction permet le lancement et le paramétrage de l'agent contrôleur d'indexation:

- Création dynamique ou suppression d'un processus d'indexation
- Réglage du temps de mise en sommeil de l'agent contrôleur d'indexation.



## Principes Fonctionnels du Moteur D'Indexation

### Index Inversé

La création et la gestion d'un **index inversé** constitue un point essentiel pour la construction d'un moteur de recherche par mot-clés efficace.

La procédure d'indexation d'un document commence d'abord par l'analyse de ce document, qui produit une liste de **signalements** décrivant le nombre d'occurrences d'un mot dans un document.

Le signalement inclut généralement les informations suivantes :

- Le **mot** lui-même
- L'**identifiant** du document
- Le ou les **emplacements des occurrences** du mot et la **fréquence** de ce mot dans le document (si possible).

Dans OneSearch, l'index de recherche se compose d'une liste de signalements triés par mot. En effet, si l'on considère les signalements comme des nuplets de forme *<Mot, Identifiant du document>*, un ensemble de documents va produire une liste de signalements triés par identifiant de document.

La construction d'un index de recherche reposant fondamentalement sur une question de tri, le choix d'une procédure de tri des signalement par mots, ou éventuellement par mots et par documents pour faciliter les recherches multi-mots, permet d'optimiser la recherche de documents contenant des mots spécifiques.

Chaque fichier d'index contient 100 mots ou entrées triées.

### Une Fonctionnalité Novatrice

Dans la plupart des moteurs de recherche, la gestion de l'index est assurée par une structure arborescente (arbre-B).

L'approche retenue pour OneSearch est légèrement différente : au lieu de gérer un index unique, OneSearch construit des segments d'index multiples et les fusionne périodiquement.

Lors de chaque indexation d'un nouveau document, OneSearch crée un nouveau segment d'index puis procède immédiatement à la fusion des petits segments d'index avec les plus grands, pour conserver un nombre total de segments réduit afin de garantir la rapidité des recherches.

Pour optimiser l'index et rendre les recherches plus rapides, OneSearch fusionne tous les segments en un seul, ce qui est particulièrement utile pour les index rarement mis à jour. Pour éviter les conflits et les verrouillages entre lecteurs et scripteurs d'index, OneSearch ne modifie jamais les segments existants mais en crée de nouveaux.

Lors de la fusion des segments, OneSearch écrit un nouveau segment et détruit les anciens. Cette approche hiérarchisée offre un haut degré de flexibilité qui équilibre la vitesse d'indexation et la vitesse de recherche. Un segment d'index est constitué de plusieurs fichiers :

- Un **index du dictionnaire** qui contient une entrée pour cent entrées du dictionnaire
- Un **dictionnaire** qui contient une entrée pour chaque mot unique
- Un **fichier de signalements** contenant une entrée pour chaque signalement.

Les segments d'index existants n'étant jamais mis à jour, il est possible de les stocker dans des fichiers plats plutôt que dans des structures arborescentes complexes.

## Le Moteur de Recherche

Le moteur de recherche de OneSearch utilise à la fois un ensemble complet de principes de construction de requêtes pour les requêtes complexes et un analyseur de requêtes intégré très simple d'utilisation, qui permet le lancement de requêtes génériques communes du type de celles décrites ci-dessous.

Les outils de recherche proposés sont entre autres : troncatures multiples, opérateurs booléens, opérateurs de comparaison, opérateurs de proximité, visualisation des index, thésaurus, extensions linguistiques. L'interrogation multi-critères peut s'effectuer champ par champ, sur un groupe de champs, sur l'ensemble des champs de la table y compris le texte intégral des documents.

### ▪ Recherche par mot-clé :

auteur: Eckel java

Recherche des documents contenant « Eckel » dans le champs *auteur* et « java » dans le champs *contenu*

### ▪ Recherche par plage de dates :

FROM dateA TO dateB

Recherche des documents dont le champs *date* contient une valeur égale ou comprise entre « dateA » et « dateB »

=JJ/MM/AAAA

<=JJ/MM/AAAA

>=JJ/MM/AAAA

### ▪ Recherche d'une chaîne de caractères :

« traitement et stockage des fichiers et documents d'entreprise »

Recherche des documents contenant la chaîne de caractères spécifiée

### ▪ Recherche avec l'opérateur booléen ET ou + :

fichier ET informatique

Recherche des documents contenant le mot « fichier » et le mot « informatique »

fichier ET « traitement informatique »

Recherche des documents contenant le mot « fichier » et la chaîne de caractères « traitement informatique »

+traitement informatique

Recherche des documents contenant le mot « traitement » et prioritairement le mot « informatique »

### ▪ Recherche avec l'opérateur booléen SAUF ou - :

fichier - traitement

Recherche du mot « fichier » en écartant les documents contenant le mot « traitement »

- **Recherche avec l'opérateur booléen OU :**

fichier traitement

Recherche des documents contenant soit le mot « fichier » soit le mot « traitement »

- **Recherche avec les opérateurs de proximité NEAR TO (adjacence) et NEXT TO (voisinage), c'est à dire recherche de deux mots voisins ou séparés par un mot ou une chaîne de caractères :**

stockage NEAR TO fichier

Recherche des documents contenant le mot « stockage » voisin du mot « fichier »

fichier *traitement des données*  
stockage

Recherche des mots « fichier » et « stockage » séparés par la chaîne de caractères « *traitement des données* »

- **Recherche par troncature** utilisant le caractère générique \* (astérisque), c'est à dire recherche commençant, finissant ou incluant une chaîne de caractère ou un caractère spécifique.

Ce mode de recherche particulier implémenté par OneSearch exploite une base de données SQL réalisant l'indexation des mots et de leurs inverses. Cette base contient, pour chaque lettre de l'alphabet d'une langue, une table des mots indexés et une table de l'inverse des mots indexés.

**a\***

Recherche des documents contenant des mots commençant par la lettre « a »

**\*ent**

Recherche des documents contenant des mots finissant par la chaîne de caractères « ent »

**p\*a**

Recherche des documents contenant des mots commençant par la lettre « p » et finissant par la lettre « a »

**\*ati\***

Recherche des documents contenant la chaîne de caractères « ati »

## Gestion de Tables de Correspondance (Mapping)

La gestion de tables de correspondance entre les lettres composant les jeux de caractères relatifs à deux langues différentes permet au moteur OneSearch d'effectuer l'indexation et la recherche de documents appartenant virtuellement à toutes les langues du monde.

La fonction Mapping permet la construction d'une table de correspondance entre un alphabet utilisant un jeu de caractères latins et un alphabet utilisant un autre jeu de caractères : arabes, cyrilliques, grecs, etc.

## Principes Fonctionnels du Moteur de Recherche

### Fonction d'Extension Linguistique (Indexation par Radicaux)

Pour la langue anglaise, OneSearch implémente l'**Algorithme de Porter** d'indexation par radicaux (réduction d'un mot à son radical), qui permet d'étendre la recherche sur un mot au radical et autres formes lexicales de ce mot.

Une requête portant sur un mot donné peut ainsi correspondre à d'autres mots similaires ou de la même famille, ce qui permet d'augmenter la portée d'une recherche

### Indexation Incrémentale et Indexation par Lots

Le moteur de recherche OneSearch supporte deux types d'indexation : indexation incrémentale et indexation par lots.

Certains moteurs de recherche ne gèrent que l'indexation par lots : une fois qu'un index a été créé pour un ensemble de documents, l'ajout de documents nouveaux devient difficile si l'on ne procède pas à la réindexation de l'ensemble des documents.

L'indexation incrémentale facilite la procédure d'ajout de nouveaux documents à un index existant, ce qui constitue un point crucial dans le cas d'applications gérant des sources de données actives, par exemple.

### Sources des Données

Contrairement à certains moteurs de recherche qui peuvent indexer uniquement des fichiers texte ou des pages Web, mais sont incapables d'indexer des données provenant d'une base de données ou de gérer des fichiers contenant des documents virtuels multiples (archives de fichiers Zip par exemple), OneSearch permet de gérer les données en faisant abstraction de la source de ces données, à condition de fournir les lecteurs de données appropriés.

### Contrôles de l'Indexation

Certains moteurs de recherche effectuent un balayage automatique d'une arborescence de répertoire ou d'un site Web pour y rechercher des documents à indexer : pourtant, les indexeurs à moteur de balayage n'offrent pas toute la souplesse requise pour les applications qui requièrent un contrôle plus fin sur les documents indexés.

Le moteur de recherche OneSearch fonctionnant principalement en mode incrémental, il permet à une application de procéder par elle-même à la recherche et à l'extraction des documents.

### Format des Fichiers

Contrairement à certains moteurs de recherche qui peuvent indexer uniquement du texte ou des documents HTML, OneSearch intègre un mécanisme de filtres qui permet d'indexer aisément des fichiers traitement de texte, des documents SGML et d'autres formats de fichiers.

### Marquage du Contenu

Plutôt que de traiter le document comme un simple flot d'unités lexicales, OneSearch permet de spécifier pour chaque document des champs d'information multiples (« sujet », « résumé », « auteur », « contenu », etc.) enrichissant la qualité sémantique de la recherche, par exemple : « auteur *contient* Molière et contenu *contient* femmes ».

OneSearch gère le marquage du contenu en traitant chaque document comme un ensemble de champs et supporte les requêtes qui spécifient le ou les champs sur lesquels porteront la recherche.

## **Gestion des Accès Simultanés**

OneSearch gère les accès transactionnels au moteur de recherche et d'indexation : plusieurs utilisateurs peuvent effectuer simultanément une recherche dans l'index, même pendant d'autres utilisateurs effectuent la mise à jour de l'index.

## Schéma de l'architecture fonctionnelle de OneSearch

### FONCTIONS DE PARAMETRAGE

Création/Vidage du Fichier d'Index

Définition des Options de Paramétrage du Système de Fichier

Définition/Suppression des Extensions de Fichiers gérées

Définition/Suppression des Langues gérées  
(Tables de correspondance et Fichier des mots blancs)

### FONCTIONS D'EXPLOITATION

Lancement/Suppression d'un Processus d'Indexation  
sur un Dossier de Fichiers

Listage des Dossiers de Fichiers Indexés

Lancement de Requêtes sur les dossiers de fichiers indexés

Accès au Fichier d'Aide

## Fonctions de Paramétrage

### Fonction SETTINGS (PARAMETRAGE)

Cette fonction assure la définition des options de paramétrage du système de fichier de OneSearch :

- Définition des chemins d'accès et des emplacements de stockage des fichiers gérés par le système (Index, fichier d'aide, base SQL, etc...)
- Définition des noms d'utilisateurs et mots de passe pour l'accès à la base SQL
- Définition des adresses de stockage des fichiers
- Définition des paramètres des documents gérés dans les fichiers alias (noms/valeurs)
- Définition des champs des documents gérés dans les fichiers alias (noms/valeurs)
- Choix de l'icône correspondant à l'application OneSearch.

Une fenêtre affiche le contenu options par défaut stockées dans le fichier de paramétrage. Les modifications (ajouts, suppression, mises à jour) sont effectuées manuellement.

Le bouton Sauvegarder permet de sauvegarder les modifications apportées au fichier de paramétrage.

Le bouton Recharger permet de recharger le contenu du fichier de paramétrage précédent.

### Fonction EXTENSIONS

Cette fonction permet de définir ou modifier la liste des formats de fichiers gérés par le moteur d'indexation OneSearch.

Une fenêtre affiche la liste des extensions de fichiers définies et reconnues par OneSearch (doc, rtf, xls, wks, html, etc.).

Les options **Ajout Extension** et **Suppression Extension** permettent d'ajouter une nouvelle extension de fichier au système ou de supprimer un format de fichier de la liste des extensions, en spécifiant le nom de l'extension à gérer et en cliquant sur le bouton **Ajouter** ou sur le bouton **Supprimer**.

### Fonction LANGUAGES (LANGUES)

Cette fonction permet de définir la liste des langues, c'est à dire des jeux de caractères reconnus comme langues d'indexation par OneSearch et correspondant aux jeux de caractères installés au niveau du système d'exploitation.

Cette fonction permet également de construire pour chaque jeu de caractères non latins une table de correspondance (*mapping*) entre chaque lettre du jeu de caractères de référence (alphabet à caractères latins) et chaque lettre du jeu de caractères non latins qui doit être reconnu par le moteur d'indexation et de recherche OneSearch.

OneSearch permet de faire correspondre à chaque langue gérée un fichier des mots blancs (*stop words*), dont le contenu peut être visualisé et modifié.

Une fenêtre affiche le contenu du dossier les langues gérées, c'est à dire la liste des jeux de caractères définis comme langues d'indexation pour OneSearch.

L'ajout ou la suppression d'une langue d'indexation s'effectuent en spécifiant le nom de la langue à ajouter ou à supprimer dans le champs **Nom Langue** et en indiquant éventuellement le chemin d'accès du fichier des mots blancs relatif à cette langue dans le champs **Fichier des Mots Blancs**, puis en cliquant sur le bouton Ajouter ou Supprimer.

La fenêtre intermédiaire ou fenêtre de mapping permet de construire la table de correspondance entre chaque lettre du jeu de caractères non latins, (saisie dans la colonne **Lettre**) et une lettre ou un phonème du jeu de caractères latins de référence (saisie dans la colonne **Mapping**).

La fenêtre inférieure **Fichier des Mots Blancs** visualise la liste des mots composant le fichier des mots blancs correspondant à la langue sélectionnée pour contrôle, enrichissement ou mise à jour.

Les boutons Sauvegarder permettent d'enregistrer les modifications apportées à la table de correspondance et au fichier des mots blancs courants ou de valider la création d'une nouvelle table de correspondance ou d'un nouveau fichier des mots blancs .

## Fonction FOLDERS (INDEXATION DOSSIERS)

Cette fonction permet la création et le lancement d'un processus élémentaire d'indexation sur un dossier de fichiers ou la suppression d'un processus d'indexation déjà lancé :

- Option **Ajouter Dossier** : spécification du chemin d'accès du dossier de fichiers à indexer puis validation par le bouton Ajouter.
- Option **Supprimer Dossier** : spécification du chemin d'accès du dossier de fichiers dont l'indexation doit être interrompue puis validation par le bouton Supprimer.

Cette fonction permet également :

- De spécifier un nouveau nom dossier pour le stockage de l'index en indiquant son chemin d'accès dans le champs **Création Nouveau Dossier Index** et en validant cette création en cliquant sur le bouton **Créer**.  
Si le nom de dossier spécifié n'existe pas, un nouveau dossier d'index est créé. Si le nom de dossier spécifié existe déjà, une fenêtre de confirmation permet la suppression du contenu de ce dossier.
- De déterminer la longueur de la période d'inactivité du processus d'indexation après la réalisation d'une procédure complète d'indexation : le moteur d'indexation se met en sommeil et scrute régulièrement l'état des demandes de lancement d'indexation (longueur exprimée en **Heures**, **Minutes** et **Secondes**).



## Fonctions d'Exploitation

### Fonction INDEXED FILES (FICHIERS INDEXES)

Cette fonction permet de contrôler et de visualiser fichier par fichier les dossiers de fichiers ayant subi avec succès une procédure d'indexation complète.

La fenêtre des dossiers indexés affiche la liste des dossiers de fichiers Alias correspondant à chaque dossier de documents indexés, classés par ordre chronologique d'indexation.

La fenêtre d'affichage des documents indexés affiche la liste des fichiers dont l'indexation s'est correctement déroulée avec l'ensemble de leurs paramètres d'indexation :

- Identifiant du document : attribué par OneSearch lors de la procédure d'indexation
- Chemin d'accès, nom et extension du document
- Type d'action effectuée pour ce document : ajout, suppression ou mise à jour du document dans l'index
- Temps d'indexation du document exprimé en millisecondes
- Nom du dossier de fichiers Alias auquel appartient le document.

Chaque colonne de la fenêtre d'affichage des documents indexés peut être déplacée ou élargie selon les besoins.

La fenêtre d'affichage des documents indexés permet de visualiser 3000 noms de fichiers indexés, avant d'être remise à jour.

Ajouter :

Affichage des paramètres du fichier en cours d'indexation.

Une fenêtre d'affichage de la liste des fichiers qui n'ont pas pu être indexés.

### Fonction SEARCH (RECHERCHE)

Cette fonction permet le lancement de requêtes portant sur un dossier de fichiers indexés et l'affichage d'une liste de documents vérifiant la recherche.

Les paramètres de la recherche sont saisis dans la zone de lancement de la recherche :

- Champs **Dossier Indexé** : spécifier le nom du dossier de documents indexés sur lequel portera la recherche.
- Champs **Rechercher** : indiquer le ou les mots ou la chaîne de caractères à rechercher, en utilisant les opérateurs booléens ou le caractère joker.

Les documents correspondants aux critères de recherche s'affichent dans la zone de visualisation des résultats de la recherche :

- **L'indicateur de résultat** affiche le nombre de documents trouvés pour la recherche en cours.

- **La fenêtre de visualisation** de la liste des documents correspondants à la recherche affiche les paramètres des documents trouvés avec l'identifiant qui leur a été attribué lors de leur intégration à la bibliothèque électronique et leur chemin d'accès complet avec leur nom et leur extension.

La fenêtre de visualisation affiche la liste des 20 premiers documents correspondants à la recherche : les boutons **Suivant** et **Précédent** permettent de parcourir la liste complète des documents trouvés en affichant la liste des 20 documents suivants ou des 20 documents précédents.

Le bouton **Aller à** suivi du numéro d'identifiant affiche le document dont l'identifiant est spécifié.

### Fonction AIDE

La fonction **Aide** affiche le sommaire des rubriques d'aide liées à l'utilisation du moteur d'indexation et de recherche OneSearch.

## Beyond basic text documents

Lucene is a fine example of good object-oriented software design and architecture. A carefully crafted division of labor between the application and the search engine lies beneath its design. This transforms indexing from a monolithic process into a collection of cooperating objects, each performing a single function and operating in a single domain. For example, when indexing a file, the `FileInputStream` class retrieves the document data; the appropriate `Analyzer` transforms it into a stream of tokens; the `IndexWriter` class indexes it; and the `FSDirectory` class stores the index on disk for later retrieval. Each of these classes performs one function, and each can be easily replaced without affecting the others.

Lucene uses three major abstractions to support building text indexes: `Document`, `Analyzer`, and `Directory`. The `Document` object represents a single document, modeled as a collection of `Field` objects (name-value pairs). For each document to be indexed, the application creates a `Document` object and adds it to the index store. The `Analyzer` converts the contents of each `Field` into a sequence of tokens.

A `Token`, the basic unit of indexing in Lucene, represents a single word to be indexed after any document domain transformation -- such as stop-word elimination, stemming, filtering, term normalization, or language translation -- has been applied. The application filters undesired tokens, like stop words or portions of the input that do not need to be indexed, through the `Analyzer` class. It also modifies tokens as they are encountered in the input, to perform stemming or other term normalization. Conveniently, Lucene comes with a set of standard `Analyzer` objects for handling common transformations like word identification and stop-word elimination, so indexing simple text documents requires no additional work. If these aren't enough, the developer can provide more sophisticated analyzers.

The application provides the document data in the form of a `String` or `InputStream`, which the `Analyzer` converts to a stream of tokens. Because of this, Lucene can index data from any data source, not just files. If the documents are stored in files, use `FileInputStream` to retrieve them, as illustrated in `IndexFile.java`. If they are stored in an Oracle database, provide an `InputStream` class to retrieve them. If a document is not a text file but an HTML or XML file, for example, you can extract content by eliminating markups like HTML tags, document headers, or formatting instructions.

This can be done with a `FilterInputStream`, which would convert a document stream into a stream containing only the document's content text, and connect it to the `InputStream` that retrieves the document. So, if we wanted to index a collection of XML documents stored in an Oracle database, the resulting code would be very similar to `IndexFiles.java`. But it would use an application-provided `InputStream` class to retrieve the document from the database (instead of `FileInputStream`), as well as an application-provided `FilterInputStream` to parse the XML and extract the desired content.

Just as Lucene allows the application to control the handling of raw document data through the Analyzer and InputStream classes, it also defines an abstract class for reading and writing the index store (Directory). Lucene also provides concrete implementations of Directory for storing indexes in RAM (RAMDirectory) or in files (FSDirectory). If, for instance, you want to store the index data in a document control system or database -- or compress or encrypt the index data -- you can simply provide your own Directory class. Most users will use the provided implementations, usually the file-based implementation. But allowing the application to handle index storage enhances the package's flexibility.

Lucene's factoring leaves the application in charge of functions that it already knows about -- selecting and retrieving documents, storing the index data -- and leaves the search engine to do what it does best. However, good factoring between the component and application domains is only part of what makes a software toolkit easy to use. A useful set of default implementations for the application-domain objects is equally important. Instead of just dumping the application-domain problems in the developer's lap, Lucene provides a set of tools for solving the most common application-domain problems. This supports the design principle of commensurate effort -- the user does not have to learn much about the architecture to implement its basic functionality, but can access more advanced functionality with additional effort. The result: developers can often integrate Lucene's searching capabilities with their projects in just a few hours.



Le logiciel OneSearch repose sur les notions suivants ::

- Le Document constitue l'unité de base d'indexation et de recherche. Chaque document est composé d'un ensemble de champs. Chaque champs possède un nom et une valeur textuelle. Un champs peut être stocké avec le document, auquel cas il apparaîtra lors de chaque opération de recherche sur ce document. Il est donc recommandé d'affecter à chaque document des champs enregistrés permettant son identification unique.
- A field is a section of a Document. Each field has two parts, a name and a value. Values may be free text, provided as a String or as a Reader, or they may be atomic keywords, which are not further processed. Such keywords may be used to represent dates, urls, etc. Fields are optionally stored in the index, so that they may be returned with hits on the document.
- A Token is an occurrence of a term from the text of a field. It consists of a term's text, the start and end offset of the term in the text of the field, and a type string. The start and end offsets permit applications to re-associate a token with its source text, e.g., to display highlighted query terms in a document browser, or to show matching text fragments in a KWIC (KeyWord In Context) display, etc. The type is an interned string, assigned by a lexical analyzer (a.k.a. tokenizer), naming the lexical or syntactic class that the token belongs to. For example an end of sentence marker token might be implemented with type "eos". The default token type is "word".
- A TokenStream enumerates the sequence of tokens, either from fields of a document or from query text. This is an abstract class. Concrete subclasses are:
  - Tokenizer, a TokenStream whose input is a Reader; and
  - TokenFilter, a TokenStream whose input is another TokenStream.
- An Analyzer builds TokenStreams, which analyze text. It thus represents a policy for extracting index terms from text. Typical implementations first build a Tokenizer, which breaks the stream of characters from the Reader into raw Tokens. One or more TokenFilters may then be applied to the output of the Tokenizer.
-