Utilitaires
IBM Security AppScan Source
Version 9.0.3.7

Guide d'utilisation

Utilitaires
IBM Security AppScan Source
Version 9.0.3.7

Guide d'utilisation

Table des matières

Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	**IBM Canada**
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.

OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

France	Canada	Etats-Unis
(Pos1)		Home
Fin	Fin	End
(PgAr)		PgUp
(PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
(Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

Brevets

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

Assistance téléphonique

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

Chapitre 1. Utilitaire de génération Ounce/Make

Ounce/Make est un outil automatisant l'importation d'informations de configuration dans AppScan Source à partir d'environnements de génération utilisant un fichier `makefile`. Ounce/Make vous évite d'avoir à importer manuellement ces informations des fichiers `makefile`.

L'utilitaire Ounce/Make fournit une interface de ligne de commande pour générer des fichiers de projet AppScan Source (`.ppf`) à partir de fichiers `makefile` (un fichier `makefile` permet de générer une application ou une bibliothèque exécutable à partir de fichiers source). Les fichiers `ppf` générés contiennent toutes les informations requises par AppScan Source pour évaluer le code source que le fichier makefile correspondant est chargé de compiler. Une fois qu'Ounce/Make a généré les fichiers `.ppf`, vous pouvez importer ou ajouter les fichiers de projet dans AppScan Source.

Pour lancer l'utilitaire :

- Sur les systèmes Windows, exécutez `<install_dir>\bin\ouncemake.exe` (où `<rép_install>` représente l'emplacement de votre installation AppScan Source), par exemple sous Windows (32 bits) :

  ```
  C:\Program Files\IBM\AppScan Source\bin\ouncemake.exe
  ```

- Sur les systèmes Linux, exécutez `<install_dir>/bin/ouncemake`, par exemple :

  ```
  /opt/ibm/appscansource/bin/ouncemake
  ```

`Make` est un outil automatisant la compilation, l'édition de liens (et ainsi de suite) de programmes, tout en prenant en compte les interdépendances des modules et leurs dates de modification. `Make` lit ses instructions depuis un fichier `makefile`, lequel spécifie un ensemble de cibles de génération, les fichiers dont elles dépendent et les commandes à exécuter pour ce faire.

Exigences

Pour créer correctement des fichiers de projet AppScan Source, vous devez exécuter Ounce/Make dans un environnement adapté. La liste suivante répertorie les exigences pour une exécution correcte d'Ounce/Make. Si toutes ces exigences ne sont pas remplies, l'exécution Ounce/Make échouera.

- Le répertoire depuis lequel s'exécute Ounce/Make doit contenir un fichier makefile valide.
- L'environnement de génération doit être capable d'émettre une commande make qui aboutisse.
- Vous devez exécuter une commande make clean avant d'exécuter Ounce/Make. Vous pouvez exécuter explicitement make clean avant Ounce/Make ou l'inclure avec Ounce/Make en spécifiant l'option `- clean`.
- Les fichiers Makefile rencontrés par Ounce/Make ne peuvent pas contenir de chemins absolus codés en dur dans les circonstances suivantes :
  - Vers l'exécutable make lors de l'appel d'un autre fichier makefile :

    Par exemple, vous ne devez pas référencer le chemin `/usr/bin/make -f makefile.mk`. Dans le fichier makefile, référencez make via son exécutable ou une variable. Il peut d'agir de la *macro make*, `${MAKE}`, ou d'une autre variable spécifiée dans le fichier des propriétés.

  - Vers l'exécutable du compilateur lors d'une compilation du code source :
    Par exemple, `/usr/bin/gcc -I.. -DFOO -o myfile.o myfile.cpp`
  - Vers l'exécutable de l'éditeur de liens lors d'une liaison de fichiers objet.
    Par exemple, `/usr/bin/ld file1.o file2.o`
- Dans des instructions `#include`.

  Pour utiliser une instruction `#include`, ajoutez l'indicateur suivant en tant qu'option de la configuration :

  `--remote_root <rép distant>`

  où `<rép distant>` définit le point de montage du répertoire distant.

  Remarque : Vous ne pouvez spécifier qu'une seule fois `remote_root`. Tous les chemins codés en dur vers des instructions `#include` doivent aboutir à un seul point de montage.
- Ne spécifiez pas de macros pour les exécutables make, compiler et linker sur la ligne de commande lors d'un appel à make (par exemple, `make CC=gcc LD=ld`).

Eléments pris en charge par Ounce/Make

Ounce/Make prend en charge plusieurs plateformes d'ordinateur, compilateurs et versions de `make`.

Plateformes

Vous pouvez exécuter Ounce/Make sur toutes les plateformes prises en charge par AppScan Source.

Versions de Make

Ounce/Make prend en charge les versions make suivantes :

- make : l'utilitaire UNIX make est un outil d'ingénierie logicielle pour la gestion et la maintenance de programmes informatiques
- gmake : GNU Make est un outil qui contrôle la génération d'exécutables et d'autres fichiers non source d'un programme à partir des fichiers source du programme
- nmake : `NMAKE.exe` (l'utilitaire de maintenance de programme de Microsoft) est un outil 32 bits qui génère des projets basés sur des commandes depuis un fichier de description

Compilateurs

Ounce/Make prend en charge les compilateurs suivants :

- GCC : GNU Compiler Collection. AppScan Source prend en charge les façades C et C++
- cl : `cl.exe` est le compilateur pour Microsoft Visual C++

Fonctionnement

Ounce/Make fonctionne avec votre commande `Make` afin d'extraire de vos fichiers makefile les informations de configuration nécessaires et de créer les fichiers de projet AppScan Source appropriés (`ppf`). Pour ce faire, Ounce/Make requiert que vous fournissiez certaines informations sur l'environnement de génération, comme la version de la commande make utilisée.

Utilisez le fichier de propriétés afin de décrire l'environnement de génération (voir «Fichier de propriétés Ounce/Make», à la page 8).

Exécution d'Ounce/Make

Vous pouvez exécuter Ounce/Make depuis n'importe quel répertoire où l'exécution d'une commande make peut aboutir. Dans la plupart des cas, l'exécutable `ouncemake` n'est pas situé dans le répertoire depuis lequel il est appelé. Par conséquent, il est recommandé d'ajouter le répertoire contenant `ouncemake` à la variable d'environnement `PATH` de sorte à faciliter l'appel d'`ouncemake`.

Par exemple, si votre code source et/ou votre fichier makefile résident dans le répertoire `\development\srcdir`, pour exécuter `ouncemake` vous devez d'abord basculer vers ce répertoire :

```
%cd \development\srcdir
```

puis exécuter `ouncemake`:

```
%ouncemake <options>
```

Remarque : Pour des raisons de compatibilité ascendante, la commande `pmake` est encore acceptée. Cependant, `pmake` pourrait être retirée dans une prochaine édition.

Désignation des fichiers de projet

Ounce/Make utilise les conventions mentionnées dans cette rubrique pour la désignation des fichiers de projet AppScan Source.

- Lors de la création de fichiers de projet AppScan Source, Ounce/Make utilise le chemin relatif entre le répertoire à partir duquel `ouncemake` a été appelé et celui où `ouncemake` crée le fichier de projet.
- Le répertoire d'où vous appelez `ouncemake` devient le premier élément dans le nom de chemin.
- Des traits de soulignement remplacent tous les séparateurs de chemin tels que les barres obliques inversées (\) sous Windows et normales (/) sous Linux.
- Lorsque le nom de fichier dépasse les limites imposées par le système d'exploitation, Ounce/Make élague des éléments du chemin, à partir de la gauche, jusqu'à ce que la longueur du nom de fichier se conforme à la convention de dénomination du système.
- Lorsque vous exécutez `ouncemake` à la racine d'un système de fichiers (par exemple, `/` ou `c:\`), `ouncemake` crée un fichier de projet AppScan Source nommé `root.ppf`.

AppScan Source sauvegarde le fichier `.ppf` créé dans l'emplacement à proximité du fichier makefile que le `ppf` représente. Par exemple, si vous exécutez Ounce/Make en créant un seul fichier de projet, AppScan Source sauvegarde le fichier `ppf` dans le répertoire depuis lequel Ounce/Make a été appelé. Reportez-vous à la rubrique «Exemple 2 : Ounce/Make avec option de récursivité», à la page 16 pour déterminer les fichiers `ppf` créés en mode multi-projets.

Remarque : Si un répertoire comporte plusieurs fichiers makefile, Ounce/Make ne crée qu'un seul fichier `.ppf` dans ce répertoire.

Exemple 1

Cet exemple illustre un fichier `ppf` créé en remplaçant les séparateurs de chemin par des traits de soulignement.

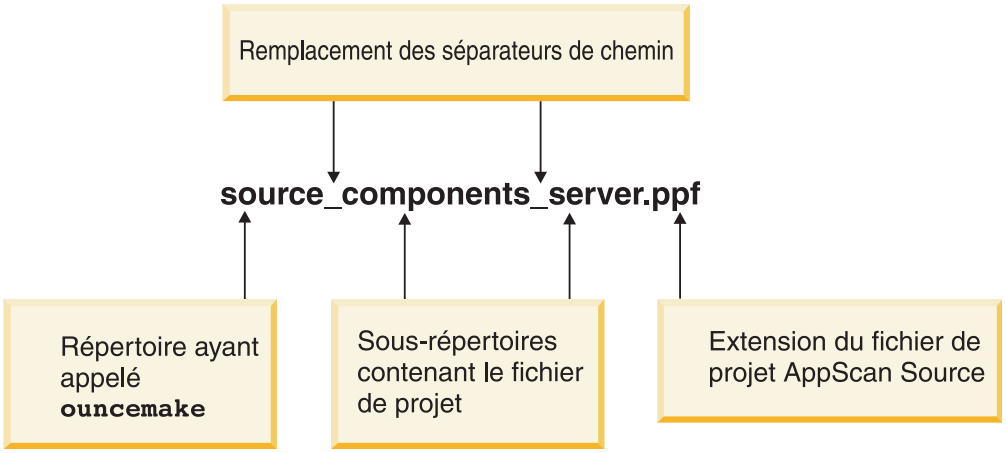
Appelez `ouncemake` depuis le répertoire suivant :

```
C:\development\source
```

Lors de son exécution, Ounce/Make crée un fichier de projet AppScan Source sous :

```
C:\development\source\components\server
```

Le nom du fichier `ppf` est `source_components_server.ppf`:



Exemple 2

Microsoft Windows et Linux limitent la longueur des noms de chemin et de fichier. Ces systèmes d'exploitation restreignent le nombre de caractères à 255. L'exemple 2 illustre le cas d'un nom de fichier dépassant les limites de longueur du chemin.

L'utilisateur appelle Ounce/Make depuis le répertoire suivant :

```
C:\path1\path2\path3\path4\path5\development\source
```

Lors de son exécution, Ounce/Make crée un fichier de projet AppScan Source dans le répertoire suivant :

```
C:\path1\path2\path3\path4\path5\development\source\components\server
```

Si le nom de fichier peut comporter au maximum 25 caractères en raison de limitations de la longueur du chemin, le nom de fichier en découlant sera alors :

```
components_server.ppf
```

Désignation explicite d'un projet

Vous pouvez éventuellement utiliser la variable d'environnement `OUNCE_PROJ_NAME` pour spécifier explicitement le nom d'un nouveau projet créé. Définissez `OUNCE_PROJ_NAME=value` comme toute autre variable d'environnement.

Si cette variable n'est pas utilisée, le nom est généré d'après la convention de dénomination existante.

Pour spécifier, par exemple, comme nom de projet `MyMakeProject` depuis la ligne de commande, entrez la commande suivante :

```
$(MAKE) $(MAKE_ARGS) all OUNCE_PROJ_NAME=MyMakeProject
```

Sortie d'Ounce/Make

Lors de son exécution, Ounce/Make génère une sortie vous avisant de certains événements.

Les événements suivants déclenchent une sortie de messages :

- Création de projet
- Appel de make
- Erreur

Création de projet

Lorsque Ounce/Make crée un fichier de projet AppScan Source, il génère un message incluant le nom du fichier `ppf`. L'exemple suivant correspond à une sortie de message de création de projet par Ounce/Make :

```
Le projet AppScan Source <nom_projet>.ppf a été créé dans le répertoire <répertoire>.
```

Appel de make

Lorsqu'Ounce/Make appelle l'exécutable make, le nom de cet exécutable et ses options sont imprimés. L'exemple suivant présente la sortie générée lorsqu'Ounce/Make appelle votre exécutable make.

```
/usr/bin/gmake -f makefile.mk release
```

Erreur

Si une erreur survient au cours de l'exécution, Ounce/Make imprime un message d'erreur la décrivant. Si l'erreur s'est produite lors de l'exécution de l'exécutable make, le message d'erreur affiché est celui de l'exécutable. Si l'erreur était spécifique à Ounce/Make, un message d'erreur approprié décrivant la condition d'erreur Ounce/Make est affiché.

Options make et syntaxe de commande Ounce/Make

Ounce/Make prend en charge de nombreuses options pouvant modifier son comportement lors de son exécution.

Vous pouvez définir ces options dans le fichier de propriétés (voir «Fichier de propriétés Ounce/Make», à la page 8 pour plus de détails) ou les inclure sur la ligne de commande. Si vous les définissez dans le fichier de propriétés, vous n'avez pas à les spécifier sur la ligne de commande chaque fois que vous exécutez Ounce/Make.

Synopsis

Ounce/Make prend en charge la syntaxe suivante :

```
ouncemake [options] [-- options_make]
```

Options

Un trait d'union (**-**) doit précéder toutes les options. Vous ne pouvez pas concaténer les options après un seul trait d'union mais devez les spécifier séparément. Par exemple, la commande :

```
ouncemake -sr
```

n'est pas une syntaxe prise en charge. Utilisez à la place :

```
ouncemake -s -r
```

Remarque : Chaque option doit être séparée par un espace.

Lorsque vous exécutez Ounce/Make, vous pouvez utiliser l'option abrégée ou le terme complet.

Les colonnes du tableau suivant décrivent chaque option.

- **Option** : identifie l'option intelligible à Ounce/Make lorsqu'elle est appelée.
- **Valeur par défaut** : le cas échéant. Explique comment Ounce/Make opère par défaut si l'option n'est pas spécifiée.
- **Description** : comportement d'Ounce/Make si cette option est utilisée.

Option	Valeur par défaut si l'option n'est pas spécifiée	Description
`-a <application_name>` `-application <application_name>`	Désactivé	Lorsque cette option est spécifiée, Ounce/Make crée un fichier d'application nommé `<application_name>.paf` contenant tous les projets créés par Ounce/Make. Le fichier est créé dans le répertoire dans lequel s'est exécuté `ouncemake`.
`-b` `-build`	Désactivé	Exécute la génération en collectant les options make. Cette option est incompatible avec Cygwin.
`-r` `-recursive`	Non récursif	Ounce/Make suit récursivement tous les appels à d'autres fichiers makefile. Par exemple, si un fichier makefile existe à la racine de l'arborescence du code source et appelle tous les fichiers makefile des sous-répertoires, l'appel `ouncemake -r` depuis le répertoire contenant le fichier makefile racine entraîne le suivi par Ounce/Make des appels aux fichiers makefile des sous-répertoires.
`-nr` `-non_recursive`	Non récursif	Ounce/Make ne suit pas récursivement tous les appels à d'autres fichiers makefile.

Option	Valeur par défaut si l'option n'est pas spécifiée	Description
-s -single_project	Mode multi-projets	Mode projet unique. En mode projet unique, Ounce/Make génère un seul fichier de projet dans le répertoire depuis lequel il a été appelé. Si cette option n'est pas spécifiée, Ounce/Make opère en mode multi-projets.
-ns -non_single_project -m -multiple_project	Mode multi-projets	Mode multi-projets. Sous ce mode, Ounce/Make génère un fichier de projet AppScan Source dans chaque répertoire et pour chaque fichier makefile rencontré.
-nv -non_verbose -q -quiet	Mode non prolixe	Mode non prolixe. Ounce/Make n'émet que ses propres messages de sortie. Ounce/Make supprime la sortie de la commande make.
-v -verbose	mode prolixe (Verbose)	Mode prolixe. Ounce/Make envoie à la sortie standard la sortie de la commande make, en plus des siennes.
-l log_level	1 (désactivé)	1 à 10. 10 représente le niveau de consignation au journal le plus complet.
-c <commande_clean> -clean <commande_clean>	Désactivé	Lorsque cette option est spécifiée, Ounce/Make interprète la <commande_clean> en tant que commande et l'exécute. La <commande_clean> est celle que l'utilisateur utiliserait normalement pour un nettoyage. Par exemple, make clean est une commande de nettoyage usuelle. Notez que cette commande doit être encadrée par des guillemets. Etant donné qu'Ounce/Make requiert un nettoyage avant son exécution, si cette option n'est pas spécifiée, une invite apparaît pour vous demander si vous désirez continuer.
-nc -no_clean	Désactivé	Indique à Ounce/Make de ne pas effectuer de nettoyage et de ne pas afficher d'invite rappelant qu'un nettoyage ne sera **pas** effectué.

Option	Valeur par défaut si l'option n'est pas spécifiée	Description
`-p <fichier_propriétés>`	n/a	Permet à l'utilisateur de spécifier un fichier de propriétés à utiliser par Ounce/Make. Le `<fichier_propriétés>` doit spécifier le chemin d'accès absolu au fichier de propriétés à utiliser par Ounce/Make.

`-?`
`-h`
`-help`
	n/a	Aide des options Ounce/Make.

Fichier de propriétés Ounce/Make

Le fichier de propriétés Ounce/Make (`ouncemake_properties.xml`) est un fichier au format XML contenant des informations sur l'environnement de génération. Lors de son exécution, Ounce/Make localise ce fichier d'après le chemin de recherche ou l'option `-p`. Le fichier de propriétés doit indiquer le compilateur utilisé dans l'environnement de génération, l'éditeur de liens et les éléments make.

Pour connaître les éléments requis et facultatifs spécifiés dans le fichier de propriétés, voir «Eléments du fichier de propriétés `Ouncemake`», à la page 9.

Lors de son exécution, si vous ne spécifiez pas l'option `-p` sur la ligne de commande, Ounce/Make tente de localiser le fichier `ouncemake_properties.xml` sur son chemin de recherche (voir «Options make et syntaxe de commande Ounce/Make», à la page 5 pour plus de détails sur l'indication des options de la ligne de commande). Le chemin de recherche Ounce/Make est dépendant de la plateforme.

Chemin de recherche Microsoft Windows :

- Répertoire de travail en cours
- `<data_dir>\config` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151)

Chemin de recherche Linux :

- Répertoire de travail en cours
- `<data_dir>/config` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151)
- Répertoire de base de l'utilisateur
- `/etc`

Remarque :

- Sous Linux, si vous ne précisez pas l'option `-p` et que le fichier de propriétés n'existe pas dans le chemin de recherche, AppScan Source en crée automatiquement un dans le répertoire de travail en cours.

- A des fins de compatibilité amont, Ounce/Make reconnaît toujours le fichier `pmake_properties.xml`. Si `pmake_properties.xml` et `ouncemake_properties.xml` existent tous deux dans le même répertoire, `ouncemake_properties.xml` est prioritaire.

Eléments du fichier de propriétés `Ouncemake`

`OuncemakeProperties`, qui est l'élément racine du fichier de propriétés, contient les éléments répertoriés dans cette rubrique.

Eléments obligatoires :

- `Compiler`
- `Linker`
- `Make`

Eléments facultatifs :

- `MakeOptions`
- `Options`
- `GlobalProjectOptions`
- `FileOptions`
- `Executable`
- `MountRoot`

`Compiler`

L'élément `Compiler` spécifie l'exécutable du compilateur utilisé dans votre environnement de génération. La valeur de cet élément doit correspondre au chemin d'accès absolu d'un exécutable. Cet élément est composé d'un attribut facultatif, `macro`, spécifiant le nom de la variable hébergeant l'exécutable du compilateur lors de la génération et d'une valeur spécifique au chemin d'accès de ce compilateur. Si vous ne spécifiez pas l'attribut `macro`, Ounce/Make le définit par défaut à `CC`. Le fichier de propriétés peut comporter des éléments `Compiler` multiples, mais au moins un de ceux-ci est requis.

La valeur de l'attribut `macro` doit être unique entre tous les éléments `Compiler`. Par exemple, vous ne pouvez pas lister `CC` en tant qu'attribut `macro` plus d'une fois.

Exemple

```
<Compiler macro=CXX>/usr/bin/gcc</Compiler>
```

Description

`CXX` : vos fichiers makefile référencent ce compilateur avec la macro `CXX`.

`/usr/bin/gcc` : Indique à Ounce/Make que vous utilisez le compilateur `/usr/bin/gcc`.

`Linker`

L'élément `Linker` spécifie l'exécutable de l'éditeur de liens utilisé dans l'environnement de génération. La valeur de cet élément doit correspondre au chemin d'accès absolu d'un exécutable. Le fichier de propriétés peut comporter des éléments `Linker` multiples, mais au moins un de ceux-ci est requis.

Cet élément est composé d'un attribut, macro, spécifiant le nom de la variable hébergeant l'exécutable linker lors de la génération. Si vous ne spécifiez pas l'attribut macro, LD constitue la macro linker par défaut.

La valeur de l'attribut macro doit être unique entre tous les éléments Linker. Par exemple, vous ne pouvez pas lister LD en tant qu'attribut Linker plus d'une fois.

Exemple

```
<Linker macro=LD>/usr/bin/ld</Linker>
```

Description

LD : indique à Ounce/Make que vous utilisez la macro LD

/usr/bin/ld: indique à Ounce/Make que vous utilisez l'éditeur de liens /usr/bin/ld

Make

L'élément Make spécifie l'exécutable make utilisé dans votre environnement de génération. La valeur de cet élément doit correspondre au chemin d'accès absolu d'un exécutable. Le fichier de propriétés peut comporter des éléments Make multiples, mais au moins un de ceux-ci est requis.

Cet élément est composé d'un attribut, macro, spécifiant le nom de la variable hébergeant l'exécutable make lors de la génération. Si vous ne spécifiez pas l'attribut macro, Ounce/Make définit cet attribut à MAKE par défaut.

La valeur de l'attribut make macro doit être unique entre tous les éléments Make.

Exemple

```
<Make macro=MAKE>/usr/bin/make</Make>
```

Description

- MAKE : indique à Ounce/Make que vous utilisez la macro MAKE.
- /usr/bin/make : indique à Ounce/Make que vous utilisez la commande make /usr/bin/make

MakeOptions

L'élément MakeOptions spécifie les options à transmettre à l'exécutable make. Cet élément est facultatif et ne peut pas figurer plus d'une fois dans le fichier de propriétés.

Exemple

```
<MakeOptions>-f makefile.mk release</MakeOptions>
```

Description

Les options :

```
-f makefile.mk release
```

sont transmises à l'exécutable make.

Options

L'élément Options spécifie les options à transmettre à Ounce/Make lors de son appel.

Cet élément fournit une alternative à certaines options de ligne de commande Ounce/Make, comme décrit dans la rubrique «Options make et syntaxe de commande Ounce/Make», à la page 5.

Les attributs utilisent la syntaxe :

```
<option> = <true | false>
```

L'élément `Options` et ses attributs ne sont pas obligatoires.

L'élément `Options` peut inclure les attributs suivants :

Attribut	**Description**
`recursive` (récursif)	Valeur booléenne. True ou false. Lorsque sa valeur est true, correspond à l'option de ligne de commande `-r`.
`single_project` (projet unique)	Valeur booléenne. True ou false. Lorsque sa valeur est true, correspond à l'option de ligne de commande `-s`.
`verbose` (prolixe)	Valeur booléenne. True ou false. Lorsque sa valeur est true, correspond à l'option de ligne de commande `-v`.
`clean` (nettoyage)	Valeur de type Chaîne, encadrée par des guillemets (`"`), telle que `"make clean"`. Lorsqu'elle est spécifiée, correspond à l'option de ligne de commande `-c`. Cette valeur doit être la commande à exécuter pour effectuer le nettoyage. Par exemple, `gmake clean`.
`build` (génération)	Valeur booléenne. True ou false. Exécute la génération en collectant les options make. **Remarque :** Incompatible avec Cygwin.
`application`	Valeur de type Chaîne. Lorsqu'elle est spécifiée, correspond à l'option `-a`. La valeur spécifiée doit être le nom d'application souhaité.
`no_clean` (pas de nettoyage)	Valeur booléenne. True ou false. Indique à Ounce/Make de ne pas effectuer de nettoyage et de ne pas afficher d'invite rappelant qu'un nettoyage ne sera pas effectué.

Remarque : Ounce/Make utilise les options spécifiées dans le fichier de propriétés. Cependant, si vous exécutez Ounce/Make avec des options inclues dans la ligne de commande, ces dernières ont préséance sur celles définies dans le fichier de propriétés. Si vous exécutez Ounce/Make sans options dans la ligne de commande, Ounce/Make applique celles du fichier de propriétés.

Exemple

Exemple de ligne du fichier de propriétés utilisant tous ces attributs :

```
<Options recursive="true" single_project="false" verbose="false"
clean="nmake.exe clean" no_clean="false"></Options>
```

Description

- `recursive="true"`
- `single_project="false"` indique à Ounce/Make d'opérer en mode multi-projets. Sous ce mode, l'attribut de récursivité doit également être défini à `true`.
- `verbose="false"` désactive la prolixité.
- `clean="nmake.exe clean"` nettoie l'environnement de génération.
- `no_clean= "false"` indique à Ounce/Make d'exécuter une commande de nettoyage et supprime le message avisant de cette opération.

GlobalProjectOptions

L'élément `GlobalProjectOptions` spécifie les options au niveau du projet s'appliquant à tous les fichiers que contient le projet. L'élément `GlobalProjectOptions` et ses attributs ne sont pas obligatoires.

La liste suivante décrit les attributs de l'élément `GlobalProjectOptions` :

- `include_paths` : valeur de type chaîne. Liste de chemins include (inclusion) séparés par des points-virgules à appliquer à tous les fichiers du projet.
- `macros` : valeur de type Chaîne. Liste de macros séparées par des points-virgules à appliquer à tous les fichiers du projet.
- `compiler_options` : valeur de type Chaîne. Liste d'options de compilateur, séparées par un espace, s'appliquant à tous les fichiers du projet. Vous ne devez pas spécifier ici les chemins `include` (inclusions) et les macros.

Exemple

Exemple de ligne du fichier de propriétés utilisant tous ces attributs :

```
<GlobalProjectOptions include_paths="/usr/include;/usr/local/include"
macros="DEBUG;WIN32" compiler_options="-f non-const-strings"/>
```

Description

- `include_paths="/usr/include;/usr/local/include"` ajoute le chemin à la section Includes de la configuration du projet.
- `macros= "DEBUG;WIN32"` ajoute les macros `DEBUG` et `WIN32` à la section Macros de la configuration du projet.
- `compiler_options=` ajoute les options de compilateur à la section Options/Command Line de la configuration du projet.

FileOptions

L'élément `FileOptions` permet d'inclure des chemins, des macros et d'autres options de compilateur pour des fichiers dotés d'extensions spécifiques. Vous pouvez spécifier `FileOptions` à de multiples reprises afin de spécifier des options distinctes pour des fichiers avec des extensions différentes. Par exemple, comme illustré ci-dessous, si vous disposez d'un projet comportant à la fois des fichiers C et C++, vous pouvez créer deux éléments `FileOptions`, à savoir un pour chaque type de fichier.

La liste suivante décrit les attributs de l'élément `FileOptions` :

- extensions: valeur de type Chaîne. Liste d'extensions de fichiers séparées par un point-virgule. Chaque fichier dont l'extension correspond à celle d'une de la liste acquiert les options spécifiées par cette propriété. Si une extension de fichier s'applique à plusieurs occurrences de la propriété FileOptions, la première occurrence dans le fichier de propriétés Ounce Make est prioritaire.
- compiler_options : valeur de type Chaîne. Liste d'options de compilateur, séparés par un espace, s'appliquant à tous les fichiers portant l'extension spécifiée. Vous ne devez pas spécifier ici les chemins include (inclusions) et les macros.
- include_paths : valeur de type Chaîne. Liste de chemins include (inclusion) séparés par des points-virgules à appliquer à tous les fichiers avec l'extension spécifiée.
- macros : valeur de type Chaîne. Liste de macros séparées par des points-virgules à appliquer à tous les fichiers avec l'extension spécifiée.

Exemples

Les exemples FileOptions suivants décrivent comment configurer le fichier de propriétés Ounce Make afin d'appliquer les options correctes aux fichiers C et C++.

L'élément FileOptions avec extensions="c" applique ses autres valeurs d'attribut uniquement aux fichiers avec une extension c de type <nom_fichier.c>. L'élément FileOptions avec extensions="cpp;cxx" appliquera ses autres valeurs d'attribut uniquement aux fichiers avec des extensions cpp (<nom_fichier.cpp>) ou cxx (<nom_fichier.cxx>).

```
<!-- g++ options for C files -->
<FileOptions
 extensions="c"
 compiler_options="-gcc_linux_i386"
 include_paths="/usr/local/include;
 /usr/lib/gcc-lib/i386-redhat-linux/3.2.3/include;
 /usr/include"
 macros=""/>

<!-- g++ options for C++ files -->
 <FileOptions
 extensions="cpp;cxx"
 compiler_options="-g++_linux_i386"
 include_paths="/usr/include/c++/3.2.3;
 /usr/include/c++/3.2.3/i386-redhat-linux;
 /usr/include/c++/3.2.3/backward;/usr/local/include;
 /usr/lib/gcc-lib/i386-redhat-linux/3.2.3/include;
 /usr/include"
 macros="__GNUG__=3" />
```

Description

extensions="c" et extensions="cpp;cxx"

spécifient à quelles extensions de fichiers ces options s'appliquent.

Executable

L'élément Executable permet d'utiliser n'importe quel exécutable (autre que compiler, linker ou make) dans l'environnement de génération. La valeur de cet élément doit correspondre au chemin d'accès absolu d'un exécutable. L'élément Executable contient un attribut, macro, spécifiant le nom de la variable utilisée pour stocker le nom de l'exécutable lors de la génération. La valeur de l'attribut macro doit être unique entre tous les éléments Executable.

Le fichier de propriétés peut contenir un nombre quelconque d'éléments Executable. L'élément `Executable` est facultatif et requis uniquement si vous utilisez dan votre environnement de génération d'autres exécutables pouvant entraîner un échec d'Ounce/Make. En identifiant ces exécutables, Ounce/Make peut empêcher son échec.

Exemple

Exemple de ligne du fichier de propriétés :

```
<Executable macro=ARCHIVE>/usr/bin/ar<Executable>
```

Description

Cet exemple indique à Ounce/Make que la génération utilise l'exécutable `/usr/bin/ar` et que les fichiers référencent celui-ci avec la macro `ARCHIVE`.

`MountRoot`

L'élément `MountRoot` est requis si vous évaluez le code source depuis un système distant. `MountRoot` indique à Ounce/Make d'ajouter en préfixe le point de montage à tous les chemins d'accès absolus `include` rencontrés.

En spécifiant un élément `MountRoot`, Ounce/Make ajoute implicitement l'option `--remote_root` à la liste des options de compilateur dans le fichier `ppf` résultant.

`MountRoot` est valide uniquement sous Linux.

Exemple

Exemple de ligne du fichier de propriétés :

```
<MountRoot>/mnt/mount_point/usr/include</MountRoot>
```

Description

Si l'un des chemins include spécifiés est `/usr/include` (comme `gcc -I/usr/include ...`), Ounce/Make ajoute en préfixe de `/usr/include` le point de montage, de sorte que le chemin stocké dans le fichier `ppf` résultant soit correct (par exemple, `/mnt/mount_point/usr/include`.

Exemples de fichiers de propriétés

L'installation de AppScan Source inclut les exemples suivants de fichiers de propriétés dans le répertoire `<data_dir>\config` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151) :

- `SampleOuncemakeProperties-Windows.xml`
- `SampleOuncemakeProperties-Linux.xml`

Vous pouvez modifier ces fichiers pour créer un fichier de propriétés personnalisé. Si vous n'êtes pas familier avec le langage XML, utilisez ces exemples en tant que modèles pour créer votre fichier.

Exemples

Cette section décrit trois façons d'utiliser Ounce/Make.

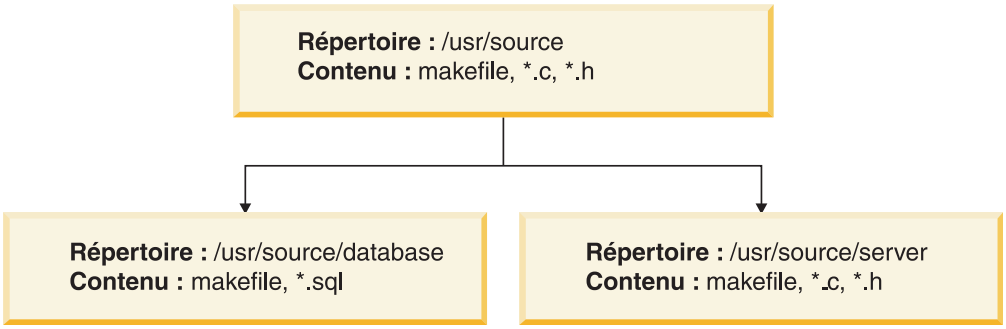
L'«Exemple 1 : Ounce/Make sans options» illustre l'utilisation d'Ounce/Make sans options, en créant un fichier de projet AppScan Source basé uniquement sur le fichier makefile dans le répertoire depuis lequel Ounce/Make est appelé.

L'«Exemple 2 : Ounce/Make avec option de récursivité», à la page 16 utilise Ounce/Make avec l'option de récursivité `-r`, laquelle indique à Ounce/Make d'opérer récursivement et de suivre tous les appels à d'autres fichiers makefile.

Dans l'«Exemple 3 : Ounce/Make avec projet unique et option de récursivité», à la page 17, Ounce/Make utilise à la fois l'option `-r` (récursivité) et `-s` (projet unique) afin de créer un fichier de projet AppScan Source unique basé sur un traitement récursif de tous les fichiers makefile rencontrés par Ounce/Make.

Structure de répertoire et de fichiers

Ces trois exemples utilisent la même structure de répertoires et de fichiers :



Ce diagramme présente un répertoire racine (`/usr/source`) contenant un fichier makefile et des fichiers source. Le répertoire `/usr/source` contient deux sous-répertoires, `/usr/source/database` et `/usr/source/server`. Le répertoire `/usr/source/database` contient un fichier makefile et des fichiers SQL. Le répertoire `/usr/source/server` contient un fichier makefile et des fichiers source.

Cet exemple repose sur les hypothèses suivantes concernant les trois fichiers makefile :

- Le fichier makefile dans `/usr/source` génère les fichiers source dans `/usr/source` et appelle les fichiers makefile de `/usr/source/database` et`/usr/source/server`.
- Le fichier makefile dans `/usr/source/database` importe les fichiers SQL dans une base de données.
- Le fichier makefile dans `/usr/source/server` génère les fichiers source dans `/usr/source/server`.

Exemple 1 : Ounce/Make sans options

Cet exemple illustre une utilisation non récursive d'Ounce/Make. Sous ce mode, Ounce/Make n'examine que le fichier makefile dans le répertoire depuis lequel il est appelé. Si le fichier makefile original comporte des appels à d'autres fichiers makefiles, Ounce/Make les ignore.

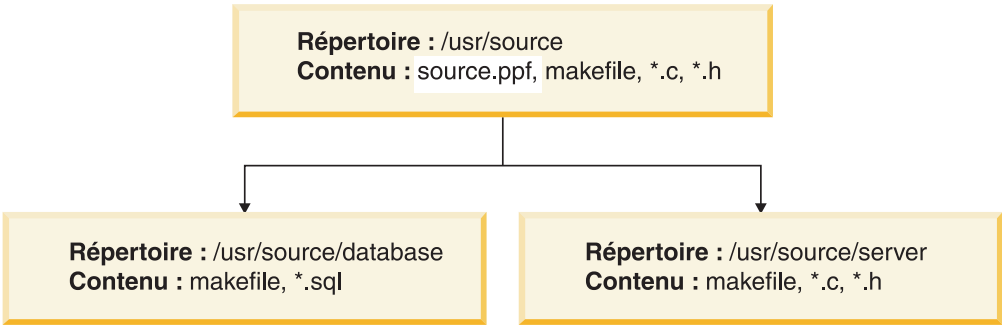
Cet exemple exécute Ounce/Make depuis `/usr/source`.

Voir «Structure de répertoire et de fichiers» pour une illustration graphique de la structure de répertoires et de fichiers sur laquelle repose cet exemple.

Commande

```
ouncemake
```

Le diagramme suivant présente le contenu des répertoires à l'issue de l'exécution d'Ounce/Make :



Après l'exécution d'Ounce/Make, `/usr/source` contient alors un fichier de projet AppScan Source nommé `source.ppf`. Ce fichier de projet héberge toutes les informations requises pour l'évaluation de tous les fichiers source dans `/usr/source`. En mode non récursif, Ounce/Make ignore dans le fichier makefile de `/usr/source` les appels aux autres fichiers makefile situés dans `/usr/source/database` et `/usr/source/server`.

Exemple 2 : Ounce/Make avec option de récursivité

Avec l'option `-r`, Ounce/Make opère en mode récursif (en suivant les appels à d'autres fichiers makefile depuis un fichier makefile). Etant donné que le mode de fichier à plusieurs projets est celui par défaut, lorsqu'il est utilisé avec l'option `-r`, Ounce/Make crée un fichier de projet AppScan Source pour chaque fichier makefile rencontré compilant le code source.

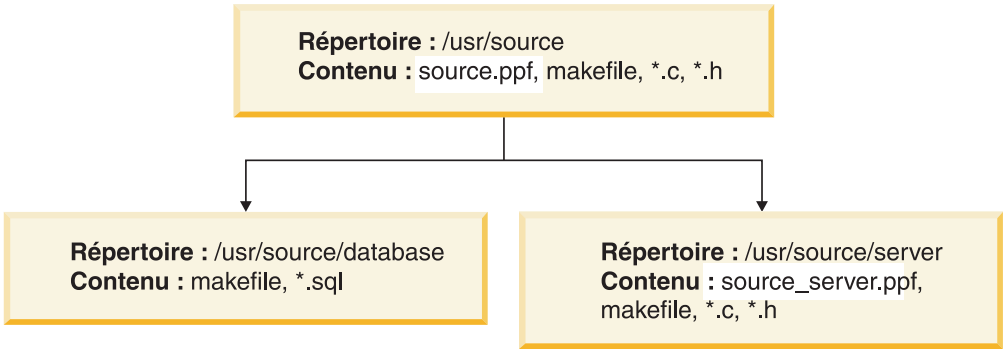
Voir «Structure de répertoire et de fichiers», à la page 15 pour une illustration graphique de la structure de répertoires et de fichiers sur laquelle repose cet exemple.

Commande

```
ouncemake -r
```

L'option `-r` (récursivité) indique à Ounce/Make de suivre les appels du fichier makefile à d'autres fichiers makefile. Pour une description plus détaillée de l'option de récursivité, reportez-vous au tableau de la rubrique «Options make et syntaxe de commande Ounce/Make», à la page 5.

Le diagramme suivant présente le contenu des répertoires à l'issue de l'exécution d'Ounce/Make :



Dans cet exemple, Ounce/Make crée un fichier de projet AppScan Source dans `/usr/source` et `/usr/source/server`. Etant donné que le fichier makefile dans `/usr/source` a appelé les fichiers dans `/usr/source/database` et `/usr/source/server`, Ounce/Make a vérifié si ces derniers ont compilé un code source.

Dans le cas du fichier makefile du répertoire `/usr/source/database`, Ounce/Make a déterminé qu'il ne compile pas de code source ; par conséquent, il n'a pas créé de fichier de projet AppScan Source. En revanche, Ounce/Make a déterminé que le fichier makefile du répertoire `/usr/source/server` compile les fichiers source de ce répertoire et an donc généré un fichier de projet AppScan Source pour les fichiers source dans `/usr/source/server`.

Exemple 3 : Ounce/Make avec projet unique et option de récursivité

L'exemple 3 illustre une utilisation récursive d'Ounce/Make en mode de projet unique. Ounce/Make génère un seul fichier de projet AppScan Source combinant le code source compilé par tous les fichiers makefile rencontrés.

Voir «Structure de répertoire et de fichiers», à la page 15 pour une illustration graphique de la structure de répertoires et de fichiers sur laquelle repose cet exemple.

Exécutez la commande suivante depuis le répertoire `/usr/source` :

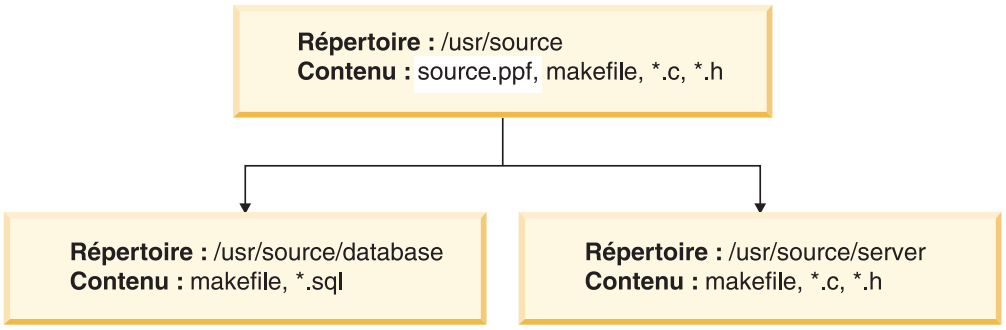
Commande

```
ouncemake -r -s
```

L'option `-r` (récursivité) indique à Ounce/Make de suivre les appels du fichier makefile à d'autres fichiers makefile. Pour une description plus détaillée de l'option de récursivité, reportez-vous au tableau de la rubrique «Options make et syntaxe de commande Ounce/Make», à la page 5.

L'option `-s` indique à Ounce/Make de générer un seul fichier de projet AppScan Source dans le répertoire depuis lequel il a été appelé, et non pas de créer un nouveau projet pour chaque fichier makefile rencontré.

Le diagramme suivant présente le contenu des répertoires à l'issue de l'exécution d'Ounce/Make.



Un seul fichier de projet AppScan Source existe sous `/usr/source`. Ce fichier de projet AppScan Source contient les informations de configuration de l'ensemble du code source sous les répertoires `/usr/source` et `/usr/source/server`.

Chapitre 2. interface de ligne de commande (CLI) d'AppScan Source

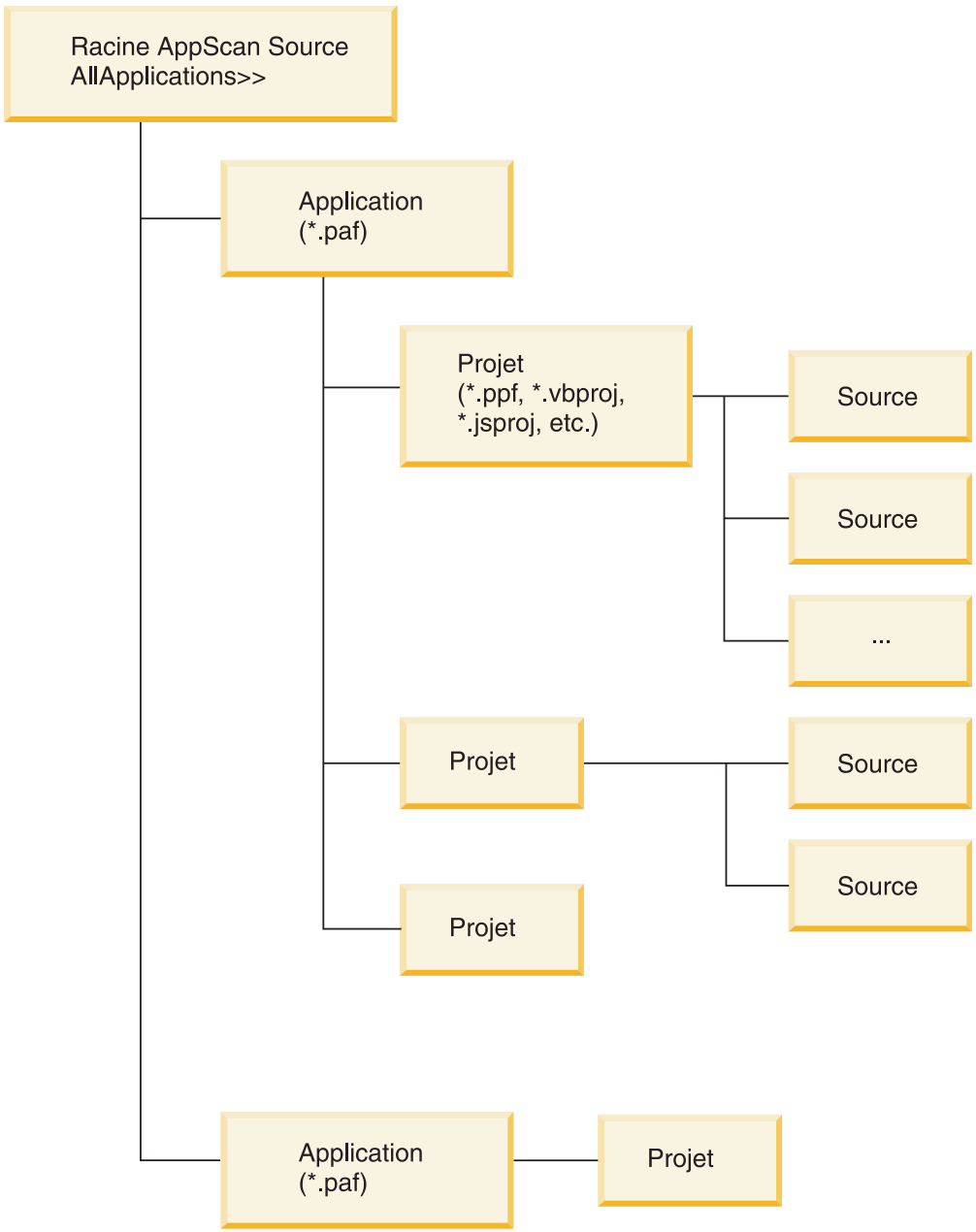
L'interface CLI est une interface de la fonctionnalité de base AppScan Source.

L'interface CLI permet l'exécution de fonctions AppScan Source à partir de la ligne de commande ou l'exécution de fichiers script pour enregistrer des commandes émises au cours d'une session interactive. Lors de l'exécution, l'interface CLI fournit à l'utilisateur des commentaires en retour sous la forme d'une sortie sur la console ou éventuellement dans un fichier journal.

Objets et contexte

Arborescence des objets AppScan Source

L'arborescence d'objets présente la structure de base des objets AppScan Source. Vous pouvez examiner n'importe quel objet (applications, projets ou fichiers source) figurant dans l'arborescence.



Objets actifs et contexte en cours

De nombreuses commandes de l'interface de ligne de commande (CLI) d'AppScan Source opèrent uniquement au sein du contexte en cours, à savoir, dans le répertoire en cours ou sur le fichier spécifié. Utilisez la commande `SetCurrentObject (SET, CD)` afin d'accéder à l'objet voulu avant d'appeler une commande opérant sur le contexte en cours.

De plus, certaines commandes de l'interface de ligne de commande CLI opèrent uniquement sur un objet actif, lequel peut correspondre au fichier source, au projet ou à une application avec une évaluation actuellement sélectionnés.

Autorisation de l'interface de ligne de commande (CLI) d'AppScan Source

De nombreuses commandes CLI requièrent que vous disposiez des autorisations AppScan Source adéquates.

Si vous n'êtes pas habilité à exécuter une action, un message d'erreur s'affiche. Notez que les commandes CLI `Scan (SC)`, `Publish` et `Report (RPT)` requièrent une licence AppScan Source for Automation.

Pour plus d'informations, contactez votre administrateur.

Démarrage de l'interface de ligne de commande (CLI) d'AppScan Source

- Sur les systèmes Windows, exécutez `<install_dir>\bin\AppScanSrcCli.exe` (où `<rép_install>` représente l'emplacement de votre installation AppScan Source). Par exemple, sous Windows (32 bits) :

  ```
  C:\Program Files\IBM\AppScanSource\bin\AppScanSrcCli.exe
  ```

- Sur les systèmes Linux, exécutez `<install_dir>/bin/appscansrccli`, par exemple :

  ```
  /opt/ibm/appscansource/bin/appscansrccli
  ```

L'interface CLI permet de transmettre une commande valide sur la ligne de commande au démarrage de l'application.

Exemple :

L'interface CLI transmet la commande `script` pour exécuter une session CLI complète automatiquement.

```
<install_dir>\bin\AppScanSrcCli script myscript.txt
```

Affichage de l'interface de ligne de commande (CLI) d'AppScan Source en langues nationales

La présente rubrique décrit les étapes nécessaires pour afficher l'interface CLI en langues nationales.

Procédure

- Ouvrez `<data_dir>\config\cli.cp` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151) dans un éditeur de texte. Ajoutez la ligne suivante au fichier :

  ```
  -Duser.language=<langue>
  ```

  Où `<langue>` correspond à la langue que vous souhaitez afficher. Les valeurs suivantes sont prises en charge :

  - `de` pour l'allemand
  - `es` pour l'espagnol
  - `fr` pour le français
  - `it` pour l'italien
  - `ja` pour le japonais
  - `ko` pour le coréen

- ru pour le russe
- pt_BR ou pt-BR pour le portugais brésilien
- zh_CN ou zh-CN pour le chinois simplifié
- zh_TW ou zh-TW pour le chinois traditionnel

Enregistrez le fichier une fois que vous aurez ajouter cette ligne.

- Ouvrez <data_dir>\config\ounce.ozsettings dans un éditeur de texte. Localisez le paramètre locale dans le fichier. Ce paramètre est similaire à ceci :

```
<Setting
   name="locale"
   value=""
   default_value=""
   description="The language (and optionally the Country/Region)
       in which UI and messages should be displayed. E.g. fr for
       French, or pt-BR for Portuguese (Brazil)"
   display_name="Locale"
   type="text"
   available_values=""
   read_only="false"
   hidden="false"
   force_upgrade="false"
 />
```

Dans ce paramètre, modifiez l'attribut value de manière à lui attribuer le même paramètre de langue que celui spécifié à l'étape précédente. Par exemple, si vous avez ajouté -Duser.language=fr à l'étape précédente, modifiez l'attribut value en value="fr".

Enregistrez le fichier une fois que vous aurez modifié ce paramètre.

- Démarrez l'interface CLI (si ces modifications ont été apportées alors que l'interface CLI était déjà en cours d'exécution, vous devez la redémarrer).

Syntaxe de commande

Les commandes de l'interface de ligne de commande (CLI) d'AppScan Source se conforment à un modèle d'utilisation comprenant des arguments requis et facultatifs, à l'instar d'un shell de commande. Les commandes CLI ne sont pas sensibles à la casse et ne requièrent pas de commutateurs pour les différents arguments.

La syntaxe de toutes les commandes CLI est la suivante :

```
commande arguments_requis [arguments_facultatifs]
```

Les descriptions et l'utilisation des commandes respectent les conventions suivantes :

- **commande** : commande réelle, identifiée par une police **Courier gras**.
- Arguments requis : identifiés par une police Courier.
- Littéraux : identifiés par des *italiques*.
- Arguments facultatifs : encadrés par des crochets [ ].
- Arguments requis comportant des options : encadrés par des crochets {}.
- Certains arguments requièrent des valeurs associées (identifiées par des crochets, < >). S'ils sont utilisés sans valeur, un message d'erreur s'affiche.
- Chaque commande dispose d'un nom complet et d'une abréviation. Les abréviations figurent entre parenthèses dans l'en-tête de la commande.
- Tous les noms de commande sont composés d'un seul mot, sans espaces ou tabulations.

- Les arguments ne requièrent pas de commutateur de commande mais sont **dépendants de l'ordre**.

  Important : Il est important de vous souvenir que les arguments sont dépendants de leur ordre.

Les arguments peuvent contenir des espaces et des tabulations. Dans ce cas, vous devez les encadrer par des guillemets (" "), faute de quoi ils seront analysés comme s'il s'agissait d'arguments distincts.

Commandes de l'interface de ligne de commande (CLI) d'AppScan Source

Cette section décrit toutes les commandes disponibles CLI, les arguments obligatoires et facultatifs, ainsi que les exemples. Chaque commande correspond à une catégorie de commande spécifique comme mentionné dans la rubrique «Récapitulatif des commandes de l'interface de ligne de commande (CLI) d'AppScan Source».

Chaque commande comporte un nom complet et une abréviation. Par exemple, la commande permettant d'analyser une application, un projet ou un fichier et de créer une évaluation est `scan`, dont l'abréviation est `sc`.

Récapitulatif des commandes de l'interface de ligne de commande (CLI) d'AppScan Source

Le tableau suivant recense et fournit une description de chaque commande de l'interface CLI et indique si une connexion est requise.

Tableau 1. Commandes de l'interface CLI

Commande et abréviation	Description	Connexion requise
`about (a)`	Affiche les informations de version et de copyright de l'interface de ligne de commande de AppScan Source.	
`clearcache (cc)`	Cette commande supprime le cache d'analyse de vulnérabilité et les données de signature des règles personnalisées. Après avoir ouvert une application à l'aide de la commande `openapplication (oa)`, vous pouvez vider son cache à l'aide de `clearcache`.	
`delete (del)`	Supprime de l'objet en cours un objet enfant.	Oui
`deleteassess (da)`	Cette commande a été renommée. Voir `removeassess (da)`.	
`deleteuser (du)`	Supprime un utilisateur de la base de données AppScan Source.	Oui

Tableau 1. Commandes de l'interface CLI (suite)

Commande et abréviation	Description	Connexion requise
`delvar (dv)`	Supprime une variable unique.	Oui
`details (det)`	Liste les caractéristiques d'évaluation de l'objet en cours.	Oui
`echo`	Répercute à l'écran toutes les entrées et sorties.	
`getaseinfo (gase)`	Imprime les paramètres AppScan Enterprise Server.	Oui
`help (?)`	Affiche l'aide pour toutes les commandes ou pour une seule commande.	
`import (im)`	Utilisez la commande `import` pour ajouter des projets (tels que `.ppf`) à une application existante.	Oui
`info (i)`	Liste des informations sur les valeurs et propriétés de l'objet en cours.	Oui
`list (ls, dir)`	Répertorie tous les objets figurant sous l'objet en cours dans l'arborescence. L'arborescence affiche une représentation graphique de l'ID, du nom et du type de l'objet.	Oui
`listassess (la)`	Affiche l'ID objet et la date et l'heure d'évaluation de l'objet en cours dans l'arborescence. Utilisez la commande `listassess` pour obtenir un ID à utiliser avec la commande `details`.	Oui
`listgroups (lgrp)`	Liste tous les groupes, avec leurs autorisations, et fournit une description de chaque groupe.	Oui
`listusers (lu)`	Répertorie tous les utilisateurs de AppScan Source.	Oui
`log`	Active ou désactive la consignation au journal des messages.	
`login (in)`	Etablit une connexion au serveur AppScan Enterprise Server (remplace `login_local (local)`).	

Tableau 1. Commandes de l'interface CLI (suite)

Commande et abréviation	Description	Connexion requise
`login_file`	Connectez-vous au serveur AppScan Enterprise Server à l'aide d'un fichier de jeton (les fichiers de jeton sont créés à l'aide de l'option `-persist` et de la commande CLI «`login (in)`», à la page 39 ou lors de la création de l'utilisateur AppScan Source for Automation).	
`logout (out)`	Effectue une déconnexion de AppScan Source et met fin à la session d'interface de ligne de commande interface de ligne de commande (CLI) d'AppScan Source.	Oui
`moduser (mu)`	Permet de modifier les informations utilisateur telles que les autorisations, l'ID utilisateur et le nom d'un utilisateur AppScan Source.	Oui
`newuser (nu)`	Créez un utilisateur AppScan Source (nom d'utilisateur, mot de passe et nom complet valides requis). Les utilisateurs AppScan Source peuvent exister dans le référentiel d'utilisateurs AppScan Enterprise Server et dans la base de données AppScan Source - ou, si vous avez des utilisateurs qui ne sont pas autorisés à accéder au serveur, ils peuvent être créés localement en tant qu'utilisateurs de AppScan Source. Vous pouvez également créer un utilisateur AppScan Source qui existe déjà sur le serveur AppScan Enterprise Server.	Oui
`openapplication (oa)`	Cette commande peut être utilisée pour ouvrir une application existante ou pour créer un fichier d'application AppScan Source.	Oui
`openassessmentfile (oaf)`	Ouvre un fichier d'évaluation AppScan Source (`nom_fichier.ozasmt`).	Oui

Tableau 1. Commandes de l'interface CLI (suite)

Commande et abréviation	Description	Connexion requise
`password (passwd)`	La commande `password` vous permet de modifier votre mot de passe ou, si vous disposez des autorisations d'administrateur, de modifier le mot de passe d'un autre utilisateur.	Oui
`printuser (pu)`	La commande `printuser (pu)` affiche les informations concernant un seul utilisateur à l'écran.	Oui
`publishassess (pa)`	Publie l'évaluation en cours ou une évaluation sélectionnée dans la base de données AppScan Source. Lorsque cette commande est utilisée, l'évaluation est rendue disponible pour un client AppScan Source tel que AppScan Source for Analysis - mais elle est indisponible pour la AppScan Enterprise Console (utilisez la commande «`publishassessase (pase)`», à la page 50 pour publier vers la AppScan Enterprise Console).	Oui
`publishassessase (pase)`	Publie l'évaluation en cours ou une évaluation sélectionnée dans AppScan Enterprise Console. Lorsque cette commande est utilisée, l'évaluation n'est pas disponible pour les clients AppScan Source tels que AppScan Source for Analysis (utilisez la commande «`publishassess (pa)`», à la page 49 pour publier vers les clients AppScan Source).	Oui
`quit`	Termine et ferme la session d'interface de ligne de commande AppScan Source. Emet une commande de déconnexion si vous êtes connecté.	
`record (rc)`	Active ou désactive l'enregistrement des commandes.	
`refresh (rf)`	`Refresh` restaure le projet ou l'application en cours à partir du disque.	Oui

Tableau 1. Commandes de l'interface CLI (suite)

Commande et abréviation	Description	Connexion requise
`register (reg)`	Enregistre des projets et des applications auprès de la base de données de AppScan Source.	Oui
`removeassess (da)`	Supprime de la mémoire l'évaluation en cours ou celle sélectionnée.	Oui
`report (rpt)`	`Report` génère un rapport AppScan Source du type spécifié, notamment des rapports sur les constatations et des rapports AppScan Source. Une licence valide pour AppScan Source for Automation est requise pour pouvoir utiliser cette commande.	Oui
`scan (sc)`	Analyse une application (ou toutes les applications), un projet ou un fichier. Une licence valide pour AppScan Source for Automation est requise pour pouvoir utiliser cette commande.	Oui
`script (scr)`	Exécute un script de commandes.	
`setaseinfo (sase)`	Spécifiez les paramètres Enterprise Console.	Oui
`setcurrentobject (set, cd)`	Utilisez la commande `setcurrentobject` pour naviguer dans l'arborescence.	Oui
`setvar (sv)`	Crée une nouvelle variable ou modifie une variable existante.	Oui
`unregister (unreg)`	Cette commande permet d'annuler l'enregistrement d'une application ou d'un projet précédemment enregistré dans le noeud en cours.	Oui

Les commandes suivantes ont été supprimées de l'interface CLI ou sont obsolètes :

- `add` : obsolète. A ne pas utiliser.
- `listproducts (lprod)` : obsolète. A ne pas utiliser.
- `liststopobject (lstop)` : obsolète. A ne pas utiliser.
- `login_admin` : supprimée. A ne pas utiliser.
- `login_local (local)` : obsolète. Utiliser `login (in)`.
- `new` : obsolète. Utiliser `openapplication (oa)`.
- `reset (r)` : supprimée. A ne pas utiliser.
- `runassess (ra)` : obsolète. Utiliser `scan (sc)`.

about (a)

Description

Affiche les informations de version et de copyright de l'interface de ligne de commande de AppScan Source.

Syntaxe

about

Exemple

```
AllApplications>> About
--------------------------------------------------------------------------------
-
|       Security AppScan Source 8.5.0.0 Build 175.
|
|       Licensed Materials - Property of IBM Corp. (C) Copyright IBM
|       Corp. and its licensors 2003, 2011.  All Rights Reserved.  IBM,
|       the IBM logo, ibm.com, Rational, AppScan and ClearQuest are
|       trademarks or registered trademarks of International Business
|       Machines Corp., registered in many jurisdictions worldwide.
|       Other product and service names might be trademarks of IBM or
|       other companies. A current list of IBM trademarks is available
|       on the Web at "Copyright and trademark information" at
|       www.ibm.com/legal/copytrade.shtml. Linux is a registered
|       trademark of Linus Torvalds in the United States, other
|       countries, or both. Microsoft, Windows, Windows NT and the
|       Windows logo are trademarks of Microsoft Corporation in the
|       United States, other countries, or both. UNIX is a registered
|       trademark of The Open Group in the United States and other
|       countries.  Java and all Java-based trademarks and logos are
|       trademarks of Sun Microsystems, Inc. in the United States, other
|       countries, or both.
|
|       This program includes: Jacorb 2.3.0, Copyright 1997-2006 The
|       JacORB project; Jericho HTML Parser 2.1, Copyright 2005 Martin
|       Jericho; and XOM 1.0d22, Copyright 2003 Elliotte Rusty Harold,
|       each of which is available under the Gnu Library General Public
|       License (LGPL), a copy of which is available in the Notices file
|       that accompanied this program.
--------------------------------------------------------------------------------
```

clearcache (cc)

Description

Cette commande supprime le cache d'analyse de vulnérabilité et les données de signature des règles personnalisées. Après avoir ouvert une application à l'aide de la commande `openapplication (oa)`, vous pouvez vider son cache à l'aide de `clearcache`.

Après avoir émis cette commande, si l'analyse incrémentielle Java est activée, le prochain examen initié pour votre application ou pour votre projet sera un examen complet.

Syntaxe

clearcache

Exemple

```
AllApplications>> openapplication SimpleIOT
AllApplications\SimpleIOT>> clearcache
AllApplications\SimpleIOT>>
```

Analyse incrémentielle pour Java

Lorsque l'analyse incrémentielle est activée, AppScan Source met en mémoire cache les données de l'analyse. Lorsque vous réanalysez le projet ou l'application, AppScan Source utilise ces données pour déterminer les modifications apportées au code, et seules les parties du code concernées par les modifications sont réanalysées. Le résultat final est une analyse complète de votre code, mais dans une fraction de temps.

Pourquoi et quand exécuter cette tâche

L'analyse incrémentielle est prise en charge sous Windows et Linux. Lorsque l'analyse incrémentielle est activée, elle est exécutée sur des projets ou des applications AppScan Source (ou des projets ou des espaces de travail Eclipse). Une fois que vous avez activé l'analyse incrémentielle, la première analyse que vous exécutez sur votre projet, application ou espace de travail est toujours une analyse intégrale (le cache d'analyse de vulnérabilité n'est mis à jour que lors d'une analyse intégrale). Cela permet à AppScan Source de mettre en cache les données des analyses suivantes. Les analyses de votre projet, application ou espace de travail ci-après sont des analyses incrémentielles, tant que le cache d'analyse de vulnérabilité n'a pas été effacé et tant que le nombre de fichiers modifiés ne dépasse par un paramètre de seuil que vous pouvez déterminer.

Pour activer et utiliser l'analyse incrémentielle, suivez cette procédure :

Procédure

1. Ouvrez `<rép_données>\config\scan.ozsettings` dans un éditeur de texte (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151). Cherchez le paramètre `incremental_analysis` dans le fichier. Ce paramètre est similaire à ce qui suit :

   ```
   <Setting
     name="incremental_analysis"
     read_only="false"
     default_value="false"
     description="Attempt to scan only changed files,
       instead of re-scanning everything."
     type="bool"
     value="false"
     display_name="Incremental Analysis"
     hidden="true"
   />
   ```

   Dans ce paramètre, modifiez l'attribut `value`. Si l'attribut est défini sur la valeur `true`, ce paramètre sera activé. S'il est défini sur `false`, AppScan Source n'effectuera pas d'analyse incrémentielle lors de l'analyse.

2. Dans `<rép_données>\config\scan.ozsettings`, cherchez le paramètre `percentage_of_files_changed` :

   ```
   <Setting
     name="percentage_of_files_changed"
     read_only="false"
     default_value="50"
     description="In incremental scanning, if percentage of files
     being changed since last scan exceeds the threshold, full
     scan will be initiated. The percentage ranges from 0 to 100.
   ```

```
  Default threshold is 50, which represents 50%."
  type="int"
  value="50"
  display_name="Percentage of files being changed"
  hidden="true"
/>
```

Ce paramètre permet de spécifier le pourcentage de fichiers à modifier avant de lancer une analyse intégrale. Par défaut, ce pourcentage de seuil est de 50 %, ce qui signifie que, si vous effectuez une nouvelle analyse après que 50 % ou plus des fichiers de votre projet, application ou espace de travail ont été modifiés, une analyse intégrale sera lancée à la place d'une analyse incrémentielle. Dans ce paramètre, modifiez l'attribut `value`, au besoin, pour utiliser le pourcentage de seuil de votre choix.

3. Enregistrez `<rép_données>\config\scan.ozsettings` après avoir modifié tous les paramètres correspondants, puis lancez ou relancez votre produit AppScan Source, qui prend en charge l'analyse incrémentielle. Par exemple, relancez AppScan Source for Analysis, the Plug-in Eclipse AppScan Source for Development ou interface de ligne de commande (CLI) d'AppScan Source, ou relancez le service AppScan Source for Automation.
4. A présent, lorsque vous réanalyserez des applications ou des projets Java avec la même configuration d'analyse, l'analyse incrémentielle sera effectuée si le nombre de fichiers modifiés ne dépasse pas le seuil et si le cache d'analyse de vulnérabilité n'a pas été effacé.
5. **Effacement du cache d'analyse de vulnérabilité :** si l'analyse incrémentielle pose des problèmes ou si vous souhaitez effectuer une analyse intégrale lorsque l'analyse incrémentielle est activée, effacez le cache de vulnérabilité avant la nouvelle analyse :
   - AppScan Source for Analysis:
     a. Ouvrez la vue Propriétés de votre projet AppScan Source. Si vous analysez une application, ouvrez la vue Propriétés pour les projets enfants (la suppression du cache d'un projet supprime également le cache de son application).
     b. Sous l'onglet Présentation, cliquez sur **Effacer le cache**.
   - Plug-in Eclipse AppScan Source for Development : supprimez `<rép_données>\temp\<workspace>\<projet>`, où :
     – `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151.
     – `<workspace>` est le nom de l'espace de travail Eclipse dans lequel vous effectuez l'analyse. Pour effacer le cache de votre espace de travail entier, supprimez l'intégralité du répertoire `<rép_données>\temp\<workspace>`.
     – `<projet>` est le nom du projet Eclipse que vous analysez. Pour effacer le cache du projet, supprimez le répertoire `<rép_données>\temp\<workspace>\<projet>`.
   - interface de ligne de commande (CLI) d'AppScan Source : utilisez la commande `clearcache`, comme indiqué dans le document *Utilitaires IBM Security AppScan Source - Guide d'utilisation*.
   - AppScan Source for Automation : utilisez l'argument `-clearcache` de la commande `ScanApplication`, comme indiqué dans *Utilitaires IBM Security AppScan Source - Guide d'utilisation*.

Résultats

Après l'analyse dans AppScan Source for Analysis, vous pouvez utiliser la fonction **Assessment Diff** pour comparer les évaluations avant et après les modifications apportées au code.

Conseil :

- Pour forcer une analyse intégrale, désactivez l'analyse incrémentielle ou effacez le cache d'analyse de vulnérabilité.
- Lors de l'analyse incrémentielle, vous devez exécuter une analyse intégrale après avoir apporté ces modifications :
  - Modifications apportées à des règles de sécurité ou à des règles personnalisées, applicables au projet ou à l'application
  - Modifications apportées à la configuration des analyses
  - Modifications apportées aux fichiers `.ozsettings`, qui concernant les analyses
  - Modifications apportées aux propriétés d'une application ou d'un projet. Par exemple, les modifications que vous apportez dans la vue Propriétés de AppScan Source for Analysis pour toutes les applications ou une application ou un projet sélectionné.
  - Ajout d'un nouveau projet à une application ou suppression d'un projet existant
  - Exclusion de fichiers des analyses. Par exemple, dans AppScan Source for Analysis, vous pouvez choisir d'exclure un fichier de l'analyse en cliquant avec le bouton droit dans la vue de l'Explorateur et en sélectionnant **Exclure des analyses**.
- Les informations actuelles sur l'analyse incrémentielle se trouvent sous http://www.ibm.com/support/docview.wss?uid=swg21994390.

Remarque :

- Après un examen incrémentiel, l'emplacement des marqueurs de constatation peut être faussé.
- Les résultats du rattrapage pour lesquels il n'y a pas de trace peuvent s'afficher dans les résultats de l'analyse incrémentielle.
- Pendant les examens incrémentiels, il est impossible d'ouvrir simultanément plusieurs produits ou composants AppScan Source. De plus, un autre utilisateur ne peut pas analyser la même application ou le même projet que celui que vous analysez, en même temps et sur le même ordinateur.

delete (del)

Description

Supprime de l'objet en cours un objet enfant.

Utilisez la commande `list (ls, dir)` pour afficher les enfants de l'objet en cours.

Identifiez l'objet enfant à supprimer à l'aide de son nom ou de son ID.

Syntaxe

del {nom|*ID* ID_objet}

- `nom` : nom de l'objet enfant. Pour afficher les enfants de l'objet en cours, émettez une commande `list`.

- *ID* : littéral. Indique une suppression d'après l'ID de l'objet.
- ID_objet : argument requis en cas de suppression d'après un ID. L'ID_objet est l'ID numérique de l'enfant à supprimer.

Exemples

- Pour supprimer une application nommé App1, entrez la commande suivante :
  ```
  AllApplications>> Delete App1
  ```
- Pour supprimer un projet portant l'ID 40, entrez la commande suivante :
  ```
  AllApplications\App1>> Delete id 40
  ```

deleteassess (da)

Cette commande a été renommée. Voir removeassess (da).

deleteuser (du)

Description

Supprime un utilisateur de la base de données AppScan Source.

Syntaxe

du nom_utilisateur

nom_utilisateur : nom de l'utilisateur AppScan Source à supprimer.

Exemple

Pour supprimer l'utilisateur agraham, entrez la commande suivante :

```
AllApplications>> deleteuser agraham
AllApplications>> Utilisateur 'agraham' supprimé.
```

delvar (dv)

Description

Supprime une variable unique.

Syntaxe

delvar <nom_variable>

nom_variable : nom de la variable à supprimer. Si une variable portant ce nom existe, elle est immédiatement supprimée, suivi d'un message de réussite de l'opération.

Exemple

Pour supprimer la variable myvar, entrez la commande suivante :

```
AllApplications>> delvar myvar
la variable '%myvar%' a été supprimée.
```

details (det)

Description

Liste les caractéristiques d'évaluation de l'objet en cours.

Extrait les caractéristiques de l'évaluation en cours ou de celle sélectionnée d'après son ID d'évaluation. Utilisez la commande `listassess (la)` pour obtenir la liste des évaluations et leurs ID.

Conseil : Utilisez la commande `log` pour activer la consignation avant d'utiliser la commande `details`. Vous pouvez générer un fichier journal au format `.csv` que vous pourrez ensuite ouvrir dans Microsoft Excel ou tout autre programme acceptant des données CSV.

Syntaxe

details [ID]

ID: ID objet d'une évaluation de la session en cours. Si un ID n'a pas été spécifié, les caractéristiques de l'évaluation la plus récente sont affichées (ou envoyées à un fichier journal).

Exemple

Pour afficher les caractéristiques de l'évaluation la plus récente, entrez la commande suivante :

```
AllApplications\Myapps>> details
File, line, col, name, type, severity, confidence
C:\MyApps\WebGoat\webgoat\src\lessons\CommandInjection.java
, 119
, 0
, java.lang.Throwable.printStackTrace
, Vulnerability.Info
, 3-SeverityType_info
, 3-ConfidenceType_low
  .
  .
  .
C:\MyApps\WebGoat\webgoat\src\session\ECSFactory.java
, 625
, 0
, java.lang.String.equalsIgnoreCase
, Vulnerability.Info
, 3-SeverityType_info
, 3-ConfidenceType_low

 -------------------
 Total Call Sites: 1283
 Total Definitive Security Findings with High Severity: 10
 Total Definitive Security Findings with Medium Severity: 15
 Total Definitive Security Findings with Low Severity: 53
 Total Suspect Security Findings with High Severity: 24
 Total Suspect Security Findings with Medium Severity: 25
 Total Suspect Security Findings with Low Severity: 6
 Total Scan Coverage Findings with High Severity: 123
 Total Scan Coverage Findings with Medium Severity: 69
 Total Scan Coverage Findings with Low Severity: 56
 Total Lines: 17197
```

```
Max V-Density: 929.8482293423273
Max V/kloc: 22.155027039599933
V-Density: 929.8482293423273
V/kloc: 22.155027039599933
```

echo

Description

Répercute à l'écran toutes les entrées et sorties.

Cette commande est généralement utilisée dans un script.

Syntaxe

echo

Exemple

Pour renvoi de commentaires :

```
3/11/08 2:15 PM  AllApplications>>
3/11/08 2:16 PM  echo "ceci est un test"
3/11/08 2:16 PM  ceci est un test
```

getaseinfo (gase)

Description

Imprime les paramètres AppScan Enterprise Server.

Cette commande affiche la configuration AppScan Enterprise Server définie par la commande `setaseinfo (sase)`.

Syntaxe

getaseinfo

Remarque :

- Pour pouvoir définir l'`url`, vous devez être connecté à l'interface de ligne de commande (CLI) d'AppScan Source avec l'autorisation `Gérer les paramètres AppScan Enterprise`. Pour plus d'informations sur les comptes utilisateur et les autorisations, reportez-vous à la section *Administration de AppScan Source* dans le manuel *IBM Security AppScan Source - Guide d'installation et d'administration*.
- L'`ID utilisateur` et le `mot de passe` sont stockés sur le poste sur lequel le client AppScan Source s'exécute (par exemple, AppScan Source for Analysis), tandis que l'`url` est stockée sur Enterprise Server (qui peut se trouver sur un poste distant). Vous ne pouvez pas accéder aux informations d'`id utilisateur` et de `mot de passe` à partir du poste distant (par exemple, en émettant la commande `getaseinfo`).

Exemple

```
AllApplications>> getaseinfo

Nom d'utilisateur :        domaine\nom_utilisateur
URL :       http://serveur_ase/ase
```

help (?)

Description

Affiche l'aide pour toutes les commandes ou pour une seule commande.

L'aide de l'interface de ligne de commande (CLI) d'AppScan Source affiche la syntaxe de commande, les alias et un exemple d'utilisation pour chaque commande. L'aide est disponible pour les commandes individuelles ou vous pouvez demander la liste de toutes les commandes avec une brève présentation de chacune d'elles.

Syntaxe

```
help [commande]
```

`commande` : commande de l'interface CLI pour laquelle vous souhaitez afficher l'aide en ligne. Vous devez vous trouver à un endroit de l'arborescence où l'aide est disponible pour l'objet spécifique concerné.

Exemples

- Pour afficher l'aide pour toutes les commandes valides :

  ```
  AllApplications>> ?
  ```

- Pour afficher l'aide pour la commande `register (reg)` :

  ```
  AllApplications>> Help register
  ```

import (im)

Description

Utilisez la commande `import` pour ajouter des projets (tels que `.ppf`) à une application existante.

Lors de l'utilisation de cette commande pour ajouter un projet, les fichiers de projet suivants sont pris en charge :

- Fichiers de projet AppScan Source (`.ppf`).
- Microsoft Visual Studio C/C++ (`.vcproj`)
- Microsoft Visual Studio C# (`.csproj`)
- Microsoft Visual Studio Visual Basic (`.vbproj`)

Lors de l'importation de projets, des caractères génériques peuvent être utilisés à la place des noms de fichier réels et/ou des extensions de fichier. Vous pouvez également spécifier un argument supplémentaire pour indiquer si tous les projets situés sous le répertoire spécifié doivent être importés (importation récursive).

Syntaxe

```
import chemin [true|false]
```

- Le nombre d'arguments varie selon le type de l'objet importé.
- `chemin` : chemin complet de l'objet importé.
- `true` ou `false` : facultatif. Indique d'inclure ou d'omettre les sous-répertoires. Valeur par défaut : `false` (omission des sous-répertoires).

Exemple

```
AllApplications\testit>> import c:\testapps\java\webgoat\*.ppf
AllApplications\testit>> ls
23942: webgoat (Project [local])
```

info (i)

Description

Liste des informations sur les valeurs et propriétés de l'objet en cours.

Syntaxe

Info

Exemple

Pour lister l'ID et le nom de l'application en cours, entrez la commande :

```
AllApplications\Myapp>> Info
```

La sortie de la commande sera similaire à ceci :

```
objet : Myapp(Application)
id: 1

...
```

list (ls, dir)

Description

Répertorie tous les objets figurant sous l'objet en cours dans l'arborescence. L'arborescence affiche une représentation graphique de l'ID, du nom et du type de l'objet.

Syntaxe

list [*all*]

- `all` : facultatif. Affiche tous les objets de l'arborescence.
- Si aucun argument n'est transmis à la commande, seuls les enfants immédiats de l'objet en cours sont listés.

Exemples

- Pour répertorier les objets de l'arborescence de l'objet en cours, entrez la commande suivante :

  ```
  AllApplications>> List
  1: WebGoat_4_0 (Application [locale] [enregistrée])
  2: test_application (Application [locale] [enregistrée])
  ```

- Pour répertorier tous les objets de l'arborescence, entrez la commande suivante :

  ```
  AllApplications>> LS all

  1: WebGoat_4_0 (Application [locale] [enregistrée])
  |----198: compile (Projet [local] [enregistré])
  |----|----470: WebContent\lp\JavaScriptValidation.html (Source)
  |----|----475: WebContent\lp\RemoteAdminFlaw.html (Source)
  |----|----483: WebContent\lp\WelcomeScreeen.html (Source)
  |----|----474: WebContent\lp\ReflectedXSS.html (Source)
  |----|----477: WebContent\lp\SqlInjection.html (Source)
  . . .
  ```

listassess (la)

Description

Affiche l'ID objet et la date et l'heure d'évaluation de l'objet en cours dans l'arborescence. Utilisez la commande `listassess` pour obtenir un ID à utiliser avec la commande `details`.

Syntaxe

listassess [*all*]

`all` : facultatif. Liste toutes les évaluations présentes en mémoire.

Exemples

- Pour afficher les informations sur l'objet d'évaluation en cours, entrez la commande suivante :

```
AllApplications\WebGoat_4_0>> listassess

AllApplications\WebGoat_4_0>> 4762: WebGoat_4_0 (Application, Tue Mar 11 14:20:23
EDT 2008)
AllApplications\WebGoat_4_0>> 9033: WebGoat_4_0 (Application, Tue Mar 11 14:43:31
EDT 2008)
```

- Pour afficher toutes les informations sur l'objet d'évaluation, entrez la commande suivante :

```
AllApplications\Applications>> ListAssess all
542: putty (Application, Fri Aug 04 08:38:18 EDT 2008)
1804: JavaAny (Application, Tue Apr 01 13:17:33 EDT 2008)
2971: snare (Application, Tue Apr 01 08:42:53 EDT 2008)
4773: SimpleIOT (Project, Tue Apr 01 07:31:11 EDT 2008)
4874: JSPWiki (Application, Tue Apr 01 13:22:08 EDT 2008)
```

listgroups (lgrp)

Description

Liste tous les groupes, avec leurs autorisations, et fournit une description de chaque groupe.

Syntaxe

listgroups

Exemple

```
AllApplications>> ListGroups

----------------------------------------
| Group: APPS
| Description: Application and Project Management
| Permissions:
|  REGISTER (Register)
|  ATTRAPPLY (Apply Attributes)
|  ATTRMODIFY (Manage Attributes)
|  SCAN (Scan)
|  VIEWREGISTER (View Registered)
| Group: ASSESSMENTS
| Description: Assessment Management
| Permissions:
|  ASMNTDELETE (Delete Published Assessments)
|  ASMNTPUBLISH (Publish Assessments)
```

```
|  ASMNTSAVE (Save Assessments)
|  ASMNTVIEWPUBLISH (View Published Assessments)
| Group: KB
| Description: Knowledgebase Management
| Permissions:
|  CUSTOM (Manage Custom Rules)
|  PATTERN (Manage Patterns)
| Group: ADMIN
| Description: Administration
| Permissions:
|  ASE (Manage AppScan Enterprise Settings)
|  LDAP (Manage LDAP Settings)
|  USER (Manage Users)
----------------------------------------
```

listusers (lu)

Description

Répertorie tous les utilisateurs de AppScan Source.

Si ces informations sont disponibles, la sortie inclut l'ID utilisateur, le nom et le référentiel d'utilisateurs.

Syntaxe

listusers

Exemple

Pour afficher la liste de tous les utilisateurs de AppScan Source :

```
AllApplications>> Listusers
All Users
  admin     (myASEServer\admin)
  jsmith    (John Smith) local
  joandarcy (Joan Darcy) local
AllApplications>>
```

log

Description

Active ou désactive la consignation au journal des messages.

Consigne une session dans un fichier spécifié ou affiche le statut de consignation actuel. La session complète, y compris les sorties de commandes, est consignée dans le fichier spécifié.

Conseil : Pour consigner uniquement les entrées de commandes, utilisez la commande `record (rc)`.

Syntaxe

log {*on* `fichier`|*off*}

- L'argument `on` ou `off` est requis pour activer ou désactiver la consignation. `Log`; s'il n'est pas suivi d'un argument, identifie le statut de consignation actuel.
- `on` : requis pour activer la consignation. `On` requiert un nom de `fichier`.
- `fichier` : nom du fichier journal. Vous pouvez spécifier le type de fichier à générer (par exemple, `.txt` ou `.csv`.

- `off` : désactive la consignation de la session en cours.

Exemples

- Pour consigner la session dans un fichier intitulé `MyLogFile.txt`, entrez la commande suivante :

  ```
  AllApplications>> Log on c:\MyLogFile.txt
  ```

- Pour désactiver la consignation, entrez la commande suivante :

  ```
  AllApplications>> Log off
  ```

login (in)

Description

Etablit une connexion au serveur AppScan Enterprise Server (remplace `login_local (local)`).

Si la connexion a abouti, l'invite change en `AllApplications>>` pour indiquer que vous êtes connecté.

Syntaxe

```
login server user_id password [-persist] [-acceptssl]
```

- `server` : requis. Spécifiez l'URL de votre instance AppScan Enterprise Server. Le format de cette URL est le suivant : `http(s)://<nom_d'hôte>:<port>/ase`, où `<nom_d'hôte>` correspond au nom de la machine sur laquelle AppScan Enterprise Server est installé, et `<port>`, au port sur lequel le serveur s'exécute. Par exemple, `https://myhost.mydomain.ibm.com:9443/ase`.
- `user_id` : requis.
  - Si vous utilisez un ID utilisateur et un mot de passe comme méthode d'authentification AppScan Enterprise Server, entrez votre ID utilisateur.
  - Si votre serveur AppScan Enterprise Server est configuré pour utiliser l'authentification Windows, entrez le nom de domaine et d'utilisateur que vous utilisez pour vous connecter au serveur Enterprise Console (en les séparant par un signe \. Par exemple, `my_domain\my_username`).
  - Si votre serveur AppScan Enterprise Server est configuré avec LDAP, entrez le nom d'utilisateur que vous utilisez pour vous connecter au serveur Enterprise Console.
  - Sous Windows, si votre serveur AppScan Enterprise Server est activé pour l'authentification CAC (Common Access Card), entrez `common_name<cn_certificate>`, où `common_name` est votre nom usuel CAC et `cn_certificate` est le nom usuel de l'émetteur de votre certificat. Si votre nom usuel (`common_name`) ou votre certificat CN (`cn_certificate`) comporte des espaces, placez-les entre guillemets (`"common_name<cn_certificate>"`).
- `password` : Requis si vous utilisez un ID utilisateur et un mot de passe comme méthode d'authentification AppScan Enterprise Server. Spécifiez le mot de passe associé à votre ID utilisateur Enterprise Server.

  Remarque : Cette option n'est pas utilisée lorsque votre serveur AppScan Enterprise Server est activé pour l'authentification CAC.
- `-persist` : facultatif. Rend persistantes les données d'identification de connexion via une clé chiffrée. Si cet argument est utilisé, `login` mémorise le nom et le mot de passe de l'utilisateur dans un fichier de clés chiffré nommé `cli.token`. Le fichier `cli.token` est sauvegardé dans le dossier `.ounce` du répertoire `home` des utilisateurs.

- `-acceptssl` : facultatif. Acceptation automatique des certificats SSL. Pour plus d'informations, voir «Certificats SSL pour AppScan Enterprise Server»

  Important : Si cet argument n'est pas inclus, la connexion à ou la publication sur AppScan Enterprise Console échouera avec un message d'erreur en cas de détection d'un certificat non valide (si vous n'avez pas déjà accepté de façon définitive le certificat pendant la connexion via un autre produit client AppScan Source).

Exemples

- L'utilisateur John Smith ouvre une session sur le serveur AppScan Enterprise Server qui a été configuré par son administrateur :

  ```
  >> login https://serveur:9443/ase/ johnsmith mypassword
  La connexion a abouti.
  AllApplications>>
  ```

- Sous Windows, l'utilisateur Jane Smith se connecte à AppScan Enterprise Server, qui est activé pour l'authentification CAC. Son nom CN est `janesmith` et celui du certificat de sa société est `mydomain-co-ABC123-CO` :

  ```
  >> login https://serveur:9443/ase/ janesmith<mydomain-co-ABC123-CO>
  ```

  Une boîte de dialogue Windows Security l'invite ensuite à entrer son code confidentiel de carte CAS. Lorsque cette étape est effectuée, le ligne de commande affiche le message suivant :

  ```
  La connexion a abouti.
  AllApplications>>
  ```

Certificats SSL pour AppScan Enterprise Server

Une fois le serveur AppScan Enterprise Server installé, vous devez le configurer afin qu'il utilise un certificat SSL valide. Si vous ne le faites pas, vous recevrez un message indiquant que la connexion est non sécurisée lors de la connexion au serveur à partir de AppScan Source for Analysis ou de l'interface de ligne de commande (CLI) d'AppScan Source - ou de AppScan Source for Development sous Windows et Linux.

Emplacement de stockage du certificat SSL

Les certificats qui ont été acceptés de façon définitive sont stockés dans `<data_dir>\config\cacertspersonal` et dans `<data_dir>\config\cacertspersonal.pem` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151). Supprimez ces deux fichiers si vous ne voulez plus que les certificats soient stockés de manière permanente.

AppScan Source for Automation et validation du certificat SSL

Par défaut, les certificats sont automatiquement acceptés lors de l'utilisation de AppScan Source for Automation. Ce comportement est déterminé par le paramètre `ounceautod_accept_ssl` dans le fichier de configuration Automation Server (`<install_dir>\config\ounceautod.ozsettings` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151)). Si ce paramètre est modifié en remplaçant `value="true"` par `value="false"`, la validation SSL sera tentée et la connexion à ou la publication sur AppScan Enterprise Console échouera avec un message d'erreur en cas de détection d'un certificat non valide.

Interface de ligne de commande AppScan Source et validation du certificat SSL

Par défaut, lorsque vous utilisez la commande CLI `login`, la validation SSL sera tentée et la connexion à ou la publication sur AppScan Enterprise Console échouera avec un message d'erreur en cas de détection d'un certificat non valide (si vous n'avez pas déjà accepté de façon définitive le certificat pendant la connexion via un autre produit client AppScan Source). Ce comportement peut être modifié à l'aide du paramètre `-acceptssl` de l'option lors de l'exécution de la commande `login`. Lorsque ce paramètre est utilisé, les certificats SSL sont automatiquement acceptés.

login_file

Description

Connectez-vous au serveur AppScan Enterprise Server à l'aide d'un fichier de jeton (les fichiers de jeton sont créés à l'aide de l'option `-persist` et de la commande CLI «`login (in)`», à la page 39 ou lors de la création de l'utilisateur AppScan Source for Automation).

Si la connexion a abouti, l'invite change en `AllApplications>>` pour indiquer que vous êtes connecté.

Syntaxe

`login_file` `server token_file [-acceptssl]`

- `server` : requis. Spécifiez l'adresse URL de votre instance Enterprise Server.
- `token_file` : requis. Indiquez le chemin et le nom du fichier de jeton. Si le fichier de jeton a été créé avec la commande CLI «`login (in)`», à la page 39, ce fichier s'intitule `ouncecli.token` et se trouve dans le dossier `.ounce` du répertoire `home` des utilisateurs. Si le fichier de jeton a été créé pour AppScan Source for Automation, il s'agit du fichier `<data_dir>/config/ounceautod.token` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151).
- `-acceptssl` : facultatif. Acceptation automatique des certificats SSL. Pour plus d'informations, voir «Certificats SSL pour AppScan Enterprise Server», à la page 40

  Important : Si cet argument n'est pas inclus, la connexion à ou la publication sur AppScan Enterprise Console échouera avec un message d'erreur en cas de détection d'un certificat non valide (si vous n'avez pas déjà accepté de façon définitive le certificat pendant la connexion via un autre produit client AppScan Source).

Exemple

L'utilisateur John Smith se connecte à AppScan Enterprise Server, à l'aide d'un fichier jeton créé sous Windows à partir de l'interface CLI :

```
>> login_file https://serveur:9443/ase/
     C:\Users\Administrator\.ounce\ouncecli.token
La connexion a abouti.
AllApplications>>
```

login_local (local)

Description

Commande obsolète dans AppScan Source version 6. Utiliser `login (in)`.

logout (out)

Description

Effectue une déconnexion de AppScan Source et met fin à la session d'interface de ligne de commande interface de ligne de commande (CLI) d'AppScan Source.

Si vous n'êtes pas connecté, `logout` n'est pas une commande valide.

Syntaxe

```
logout
```

Exemple

Pour se déconnecter, entrez la commande suivante :

```
AllApplications>> Logout
>> Déconnecté.
>> Interface de ligne de commande Security AppScan Source existante...
```

moduser (mu)

Description

Permet de modifier les informations utilisateur telles que les autorisations, l'ID utilisateur et le nom d'un utilisateur AppScan Source.

Syntaxe

```
moduser --userid|-u <id util>
[--fullname|-f <prénom et nom de l'utilisateur>]
[--group [groupe[:permission[;permission...]
 [--group...]]
[--removegroup [groupe[:permission
 [;permission...] [--removegroup...]]
```

Nom d'utilisateur et nom

- `--username|-u` : requis. Nom d'utilisateur AppScan Source valide.
- `--fullname|-f` : facultatif. Nom complet de l'utilisateur. Si l'entrée inclut des espaces, encadrez-la de guillemets (") ; par exemple, `-f "Joe Smith"`.

Groupes et permissions

Facultatif.

Les groupes et autorisations identifient les tâches AppScan Source qui sont autorisées pour cet utilisateur. Les tâches non spécifiquement identifiées dans le cadre d'une permission sont disponibles pour tous les utilisateurs :

`--group` : groupes et permissions de groupes à ajouter à l'utilisateur. La spécification d'un groupe sans mention de permissions accorde à l'utilisateur toutes celles rattachées à ce groupe.

ou

`--removegroup` : groupes et permissions de groupes à retirer pour cet utilisateur. La spécification d'un groupe sans mention de permissions retire à l'utilisateur toutes celles rattachées à ce groupe.

Les groupes et permissions sont les suivants :

- `ASSESSMENTS` : permissions au niveau de l'évaluation.
  - `ASMNTDELETE` : suppression d'évaluations publiées.
  - `ASMNTPUBLISH` : publication d'évaluations.
  - `ASMNTSAVE` : enregistrement d'évaluations.
  - `ASMNTVIEWPUBLISH` : affichage d'évaluations.
- `ADMIN` : permissions d'administration.
  - `ASE` : gestion des paramètres AppScan Enterprise
  - `USER` : gestion des paramètres utilisateur, notamment l'ajout et la suppression d'utilisateurs et la modification de leurs permissions.
- `APPS` : permissions au niveau de l'application et du projet.
  - `ATTRAPPLY` : application d'attributs à des applications.
  - `ATTRMODIFY` : création, suppression et modification d'attributs.
  - `VIEWREGISTER` : affichage des applications et projets enregistrés.
  - `REGISTER` : enregistrement/annulation d'enregistrement d'applications et de projets. Implique de détenir une autorisation `VIEWREGISTER`.
  - `SCAN` : analyse d'applications et de projets.
- `KB` : autorisations de gestion de la Base de connaissances.
  - `CUSTOM` : gestion des règles personnalisées.
  - `PATTERN` : création, édition ou modification de schémas.
- `FILTER` : gestion des filtres.
  - `SHAREDFILTERS` : gestion des filtres partagés.
- `SCANCONFIG` : gestion des configurations d'examen
  - `SHAREDCONFIGS` : gestion des configurations d'examen partagées.

Exemple

Après la création des données d'identification de l'utilisatrice Joan Darcy (à l'aide de la commande `newuser (nu)`), l'administrateur système a déterminé qu'elle a uniquement besoin des autorisations d'enregistrement, de publication et d'affichage, mais pas de l'autorisation de suppression. De plus, Joan a besoin des autorisations Base de connaissances Patterns (Schémas) :

```
moduser --userid joandarcy --removegroup
ASSESSMENTS:ASMNTDELETE --group KB:PATTERN
```

newuser (nu)

Description

Créez un utilisateur AppScan Source (nom d'utilisateur, mot de passe et nom complet valides requis). Les utilisateurs AppScan Source peuvent exister dans le référentiel d'utilisateurs AppScan Enterprise Server et dans la base de données AppScan Source - ou, si vous avez des utilisateurs qui ne sont pas autorisés à accéder au serveur, ils peuvent être créés localement en tant qu'utilisateurs de AppScan Source. Vous pouvez également créer un utilisateur AppScan Source qui existe déjà sur le serveur AppScan Enterprise Server.

Remarque : La commande `newuser (nu)` ne s'applique pas si votre serveur AppScan Enterprise Server est activé pour l'authentification CAC (Common Access Card).

Syntaxe

```
newuser --userid|-u <id util>
--password|-p <mot de passe>
--fullname|-f <prénom et nom de l'utilisateur>
[--group [groupe[:permission[;permission...]
 [--group...]]
```

Informations d'identification

- `--userid|-u` : requis. ID utilisateur. Les espaces ne sont pas admis.
- `--password|-p` : mot de passe de l'utilisateur.
- `--fullname|-f` : nom complet de l'utilisateur. Si l'entrée inclut des espaces, encadrez-la de guillemets (") ; par exemple, `-f "Joe Smith"`.

Groupes et autorisations

Facultatif. Les groupes et autorisations identifient les tâches AppScan Source qui sont autorisées pour cet utilisateur. Les tâches non spécifiquement identifiées dans le cadre d'une permission sont disponibles pour tous les utilisateurs :

`--group` : groupes et permissions de groupe à ajouter à l'utilisateur. La spécification d'un groupe sans mention de permissions accorde à l'utilisateur toutes celles rattachées à ce groupe. Les groupes et leurs permissions sont les suivants :

- `ASSESSMENTS` : permissions au niveau de l'évaluation.
  - `ASMNTDELETE` : suppression d'évaluations publiées.
  - `ASMNTPUBLISH` : publication d'évaluations.
  - `ASMNTSAVE` : enregistrement d'évaluations.
  - `ASMNTVIEWPUBLISH` : affichage d'évaluations.
- `ADMIN` : permissions d'administration.
  - `ASE` : gestion des paramètres AppScan Enterprise
  - `USER` : gestion des paramètres utilisateur, notamment l'ajout et la suppression d'utilisateurs et la modification de leurs permissions.
- `APPS` : permissions au niveau de l'application et du projet.
  - `ATTRAPPLY` : application d'attributs à des applications.
  - `ATTRMODIFY` : création, suppression et modification d'attributs.
  - `VIEWREGISTER` : affichage des applications et projets enregistrés.

  - REGISTER : enregistrement/annulation d'enregistrement d'applications et de projets. Implique de détenir une autorisation VIEWREGISTER.
  - SCAN : analyse d'applications et de projets.
- KB : autorisations de gestion de la Base de connaissances.
  - CUSTOM : gestion des règles personnalisées.
  - PATTERN : création, édition ou modification de schémas.
- FILTER : gestion des filtres.
  - SHAREDFILTERS : gestion des filtres partagés.
- SCANCONFIG : gestion des configurations d'examen
  - SHAREDCONFIGS : gestion des configurations d'examen partagées.

Authentification LDAP

Vous ne pouvez pas ajouter d'utilisateurs LDAP au référentiel d'utilisateurs AppScan Source s'ils ne figurent pas déjà dans le référentiel d'utilisateurs AppScan Enterprise Server. Pour ajouter un utilisateur AppScan Source qui sera authentifié via LDAP, vous devez avoir configuré le référentiel d'utilisateurs AppScan Enterprise Server pour utiliser un référentiel d'utilisateurs LDAP. Pour plus d'informations, voir le manuel AppScan Enterprise Server *Planning & Installation Guide*.

Si vous utilisez l'authentification LDAP et que vous souhaitez ajouter un utilisateur AppScan Source qui ne fait pas partie d'un groupe d'utilisateurs LDAP, exécutez la commande newuser.

Exemple

Création d'un utilisateur nommé Joan Darcy sur le serveur AppScan Enterprise Server. Son nom d'utilisateur est joandarcy et son mot de passe, 123456. Joan peut utiliser AppScan Source avec toutes les autorisations accordées aux groupes APPS et ASSESSMENTS, ainsi qu'avec la permission d'opérer sur les règles personnalisées du groupe KB :

```
AllApplications>> newuser --userid joandarcy --password 123456
--fullname "Joan Darcy" --group APPS --group ASSESSMENTS --group KB:CUSTOM
AllApplications>> Utilisateur créé 'joandarcy'. ID utilisateur : 888
```

openapplication (oa)

Description

Cette commande peut être utilisée pour ouvrir une application existante ou pour créer un fichier d'application AppScan Source.

Lors de l'utilisation de cette commande pour ouvrir une application, les fichiers d'application suivants sont pris en charge :

- Fichiers d'application AppScan Source (.paf).
- Espaces de travail Eclipse ou Rational Application Developer for WebSphere Software (RAD) (.ewf)

  Remarque : Les fichiers .ewf sont générés lorsque vous utilisez openapplication pour ouvrir un répertoire d'espace de travail (en spécifiant son chemin d'accès).
- Fichiers WAR (.war)
- Fichiers EAR (.ear)

- Windows uniquement : fichiers Microsoft Visual C++ Workspace (`.dsw`)
- Windows uniquement : Fichiers de solution Microsoft Visual Studio.NET (`.sln`)

Remarque : Pour savoir quelles versions des fichiers importés sont prises en charge par AppScan Source for Analysis, AppScan Source for Automation et l'interface de ligne de commande d'AppScan Source, voir http://www.ibm.com/support/docview.wss?uid=swg27027486. Dans cette page, sélectionnez l'onglet de la version d'AppScan Source que vous utilisez, puis sélectionnez le composant AppScan Source que vous utilisez. Si AppScan Source prend en charge l'ouverture et l'examen de fichiers provenant d'autres environnements de développement, cette prise en charge est indiquée à la section **Compilateurs et langues** de l'onglet **Logiciels pris en charge**.

Syntaxe

```
openapplication file [-appserver_type]
[-include_all_lib_jars] [-include_lib_jars] [-no_ear_project]
```

- `file` : Requis.
  - Si vous utilisez la commande pour ouvrir une application, indiquez le chemin et le nom de fichier de l'application existante.
  - Si vous utilisez la commande pour créer une applicationAppScan Source, indiquez un chemin et un nom de fichier valides (vérifiez que le nouveau nom de fichier n'existe pas déjà).
  - Si vous utilisez la commande pour ouvrir un espace de travail Eclipse ou Rational Application Developer for WebSphere Software (RAD), il suffit de spécifier le chemin d'accès à l'espace de travail.
- `-appserver_type` : Facultatif. Si l'application en cours d'ouverture inclut JavaServer Pages (par exemple, un fichier `WAR` ou `EAR`), utilisez ce paramètre pour indiquer le serveur d'application à utiliser pour la compilation de JSP. Spécifiez l'une des options suivantes, avec des guillemets :
  - `Tomcat 7`
  - `Tomcat 8`
  - `WebSphere 7.0`
  - `WebSphere 8.0`
  - `WebSphere 8.5`
  - `WebLogic 11g`
  - `WebLogic 12c`

  Remarque :
  - Avant de spécifier un serveur d'application, vérifiez qu'il a été configuré correctement dans les préférences AppScan Source for Analysis.
  - Si `-appserver_type` n'est pas utilisé, le compilateur JSP par défaut défini actuellement dans AppScan Source for Analysis sera utilisé pour la compilation JSP. Prêt à l'emploi, Tomcat 7 est le compilateur JSP par défaut.
- Pour les fichiers `WAR` :
  - `-include_all_lib_jars` : Utilisez ce paramètre pour prendre en compte toutes les bibliothèques du fichier `WAR` lors de l'examen.
  - `-include_lib_jars` : Utilisez ce paramètre pour préciser les bibliothèques du fichier `WAR` à prendre en compte lors de l'examen. Dans ce cas, entrez les bibliothèques sans leur chemin et séparez-les par une virgule si vous en entrez plusieurs.

- `-no_ear_project` : Lors de l'importation d'un fichier `EAR`, un projet destiné au stockage des bibliothèques partagées est créé automatiquement. En l'absence de bibliothèques partagées, le fichier est quand-même créé, mais reste vide. Ce paramètre permet de ne pas créer de projet pour le fichier `EAR`.

Exemples

- Pour ouvrir un fichier de solution Microsoft Visual Studio .NET, sous Windows :
  ```
  AllApplications>> openapplication c:\mysln.sln
  ```
- Pour ouvrir un espace de travail Eclipse :
  ```
  AllApplications>> oa "C:\Users\myname\My Documents\myworkspace"
  ```
  ou
  ```
  AllApplications>> oa "C:\Users\myname\My Documents\myworkspace\myworkspace.ewf"
  ```
- Pour ouvrir un fichier `WAR` et ne prendre en compte que certaines bibliothèques lors de l'examen :
  ```
  AllApplications>> oa c:\mywar.war -include_lib_jars lib1.jar,lib2.jar
  ```

openassessmentfile (oaf)

Description

Ouvre un fichier d'évaluation AppScan Source (`nom_fichier.ozasmt`).

L'évaluation ouverte devient l'évaluation actuelle.

Syntaxe

`openassessmentfile` `chemin_fichier`

`chemin_fichier` : chemin et nom du fichier d'évaluation existant.

Exemple

Pour ouvrir une évaluation sauvegardée auparavant, entrez la commande suivante :

```
AllApplications>> OpenAssessmentFile c:\priority.ozasmt
```

password (passwd)

Description

La commande `password` vous permet de modifier votre mot de passe ou, si vous disposez des autorisations d'administrateur, de modifier le mot de passe d'un autre utilisateur.

Pour modifier un mot de passe, entrez deux fois le nouveau mot de passe. Les deux saisies du nouveau mot de passe doivent être identiques pour que la modification prenne effet. Lors de la modification de votre propre mot de passe, vous devez d'abord entrer votre mot de passe actuel.

Syntaxe

`password` [`nom_utilisateur`]

`nom_utilisateur` : nom de l'utilisateur dont le mot de passe doit être modifié. Si vous ne spécifiez pas un nom d'utilisateur, votre propre mot de passe sera modifié.

Exemples

- Pour modifier votre propre mot de passe, entrez la commande suivante :

```
AllApplications>> Password
Type current password: ******
Type new password: *******
Verify new password: *******
Password changed.
AllApplications>>
```

- Pour modifier le mot de passe d'un autre utilisateur (si vous disposez de privilèges d'administrateur), entrez la commande suivante :

```
AllApplications>> passwd johnsmith
Changing password for user johnsmith
Type new password: *******
Verify new password: *******
Password changed.
AllApplications>>
```

printuser (pu)

Description

La commande printuser (pu) affiche les informations concernant un seul utilisateur à l'écran.

Syntaxe

printuser nom_utilisateur

nom_utilisateur : nom de l'utilisateur AppScan Source.

Exemple

Affichage des données originales de l'utilisateur Joan Darcy :

```
printuser joandarcy
User: joandarcy
First Name: Joan
Last Name: Darcy
e-mail Address: jdarcy@example.com
Groups:
 - APPS     (Application and Project Management)
  REGISTER     (Register)
  ATTRAPPLY     (Apply Attributes)
  ATTRMODIFY     (Manage Attributes)
  SCAN       (Scan)
  VIEWREGISTER    (View Registered)
 -(ASSESSMENTS) (Assessment Management)
  ASMTDELETE     (Delete Published Assessments)
  ASMNTPUBLISH   (Publish Assessments)
  ASMNTSAVE    (Save Assessments)
  ASMNTVIEWPUBLISH (View Published Assessments)
 -[KB]     (Knowledgebase Management)
  CUSTOM      (Manage Custom Rules)
 -[ADMIN]    (Administration)
```

Remarque : Les crochets [ ] indiquent que l'utilisateur ne dispose pas de toutes les permissions accordées à ce groupe.

publishassess (pa)

Description

Publie l'évaluation en cours ou une évaluation sélectionnée dans la base de données AppScan Source. Lorsque cette commande est utilisée, l'évaluation est rendue disponible pour un client AppScan Source tel que AppScan Source for Analysis - mais elle est indisponible pour la AppScan Enterprise Console (utilisez la commande «`publishassessase (pase)`», à la page 50 pour publier vers la AppScan Enterprise Console).

Une licence valide pour AppScan Source for Automation est requise pour pouvoir utiliser cette commande.

Lorsqu'une évaluation est publiée à l'aide de cette commande, l'évaluation est automatiquement enregistrée sur AppScan Enterprise Server. Vous n'avez pas besoin d'enregistrer manuellement l'évaluation lorsque vous utilisez cette commande.

Syntaxe

pa [*ID*]

ID : littéral facultatif. Identifie l'ID de l'évaluation. Si aucun ID n'est spécifié, la commande s'applique à l'examen le plus récent. Vous pouvez utiliser la commande `listassess (la)` pour trouver l'ID d'évaluation.

Exemple

Dans l'exemple suivant, l'objet en cours est défini à l'application 'pebble', le projet 'pebble' est analysé et l'évaluation est ensuite publiée (et enregistrée automatiquement par la même occasion).

```
AllApplications>> cd pebble
AllApplications\pebble>> scan pebble
New Scan started

Scanning Project pebble (1 of 1)
Preparing project for scan...
  .
  .
Preparing for Vulnerability Analysis...
Performing Vulnerability Analysis...
Generating Findings...
Preparing project for scan...
  .
  .
Scanned Project pebble:
  Total files: 15
  Total findings: 167
  Total lines: 385
 vkloc: 0.44448395412925595
 v-Density: 22.446439683527426
Scanned Application pebble:
  Total files: 15
  Total findings: 167
  Total lines: 385
 vkloc: 0.44448395412925595
 v-Density: 22.446439683527426
Scan completed:
  Total files: 15
  Total findings: 167
```

```
  Total lines: 385
 vkloc: 0.44448395412925595
 v-Density: 22.446439683527426
Elapsed Time - 18 Seconds

AllApplications\pebble>> publishassess

Assessment Successfully Published.
```

publishassessase (pase)

Description

Publie l'évaluation en cours ou une évaluation sélectionnée dans AppScan Enterprise Console. Lorsque cette commande est utilisée, l'évaluation n'est pas disponible pour les clients AppScan Source tels que AppScan Source for Analysis (utilisez la commande «publishassess (pa)», à la page 49 pour publier vers les clients AppScan Source).

Syntaxe

publishassessase
-aseapplication <application_ase> [*id*] [*chemin*]
[-folder <emplacement>] [-name <nom_de_l'évaluation_publiée>] [-preventOverwrite]

- -aseapplication <application_ase> : Cette option est requise lors d'une connexion à AppScan Enterprise Server versions 9.0.3 et supérieures (sauf si vous désactivez l'exigence, comme indiqué ici). L'association à une application est facultative en cas de connexion aux versions antérieures de AppScan Enterprise Server. Utilisez cette option pour spécifier l'application Enterprise Console à associer à l'évaluation.
- *ID* : littéral facultatif. Identifie l'ID de l'évaluation. Vous pouvez utiliser la commande listassess (la) pour trouver l'ID d'évaluation.
- *chemin* : littéral facultatif. Chemin et nom du fichier d'évaluation.
- -folder <emplacement> : facultatif. Cette option s'applique uniquement en cas de connexion à des versions AppScan Enterprise Server antérieures à la version 9.0.3. Indiquez le dossier Enterprise Console dans lequel publier l'évaluation. Si cet argument n'est pas utilisé, l'évaluation sera publiée sur votre dossier par défaut Enterprise Console.
- -name <nom_évaluation_publiée> : facultatif. Nom sous lequel l'évaluation est sauvegardée dans Enterprise Console. Si cet argument n'est pas utilisé, un nom est généré en fonction de l'application AppScan Source qui a été examinée pour générer l'évaluation (ce nom est préfixé par AppScan Source :).
- -preventOverwrite : facultatif. Incluez cet argument pour annuler la publication si une évaluation portant le même nom existe déjà sur le serveur.

Lorsque l'argument facultatif est un entier, la commande suppose qu'il s'agit de l'ID de l'évaluation. Si ce n'est pas un entier, la commande suppose qu'il désigne le chemin d'un fichier d'évaluation sauvegardé.

Si aucune évaluation n'est spécifiée avec la commande id ou path, l'évaluation générée par l'examen le plus récent est utilisée.

Important :

Lors de la mise à niveau vers AppScan Source version 9.0.3.4, vous remarquerez les modifications suivantes :

- Lors de la publication d'une évaluation sur AppScan Enterprise Console, vous devez désormais associer celle-ci à une application dans AppScan Enterprise (si vous utilisez AppScan Enterprise Server version 9.0.3 ou supérieure). En conséquence, les scripts d'automatisation peuvent échouer s'ils ne contiennent pas l'association d'applications. AppScan Enterprise Server, l'association d'applications est requise pour bénéficier de ses fonctions de gestion des risques de sécurité pour les applications. Voir http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/c_overview.html.
- En outre, vous devez retirer le port de l'URL d'AppScan Enterprise.
  1. Dans AppScan Source for Analysis, cliquez sur **Editer** > **Préférences**.
  2. Dans les paramètres d'AppScan Enterprise Console, retirez le port de la zone **URL de la console Enterprise**.
- Après sa publication, l'évaluation n'est disponible que dans la vue Surveillance de AppScan Enterprise (dans les versions précédentes, elle était disponible dans les vues Examens). La migration vers cette vue est décrite à la rubrique http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/t_workflow_for_applications.html.

Il s'agit du résultat du changement de protocole de communication entre AppScan Source et AppScan Enterprise Server, requis pour la publication sur AppScan Enterprise Server lors de l'utilisation de l'authentification CAC (Common Access Card).

Si vous ne souhaitez pas publier des évaluations sur AppScan Enterprise Server lorsque l'authentification CAC est activée, ni tirer parti des fonctions de gestion des risques de sécurité pour les applications offertes par Enterprise Server, vous pouvez rétablir l'ancien protocole de communication de la manière suivante :

1. Ouvrez `<datrép_données;\config\ounce.ozsettings` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151).
2. Dans ce fichier, repérez le paramètre suivant :

```
<Setting
  name="force_ase902_assessment_publish"
  value="false"
  default_value="false"
  description="Use ASE 9.0.2-style assessment publish"
  display_name="Use ASE 9.0.2-style assessment publish"
  type="boolean"
  read_only="true"
  hidden="true"
/>
```

3. Remplacez `value="false"` par `value="true"` puis sauvegardez le fichier.
4. Redémarrez le produit AppScan Source à partir duquel vous publiez les évaluations.

Lorsque ce paramètre est défini sur `value="true"` :

- Si vous associez une évaluation à une application dans AppScan Enterprise lors de la publication, l'évaluation sera disponible dans les vues Surveillance et Examens.
- Si vous n'associez pas l'évaluation à une application lors de la publication, celle-ci sera disponible dans la vue Examens.
- Vous ne pourrez pas publier des évaluations dans AppScan Enterprise Server lorsque l'authentification CAC est activée.

Pour plus d'informations, voir http://www.ibm.com/support/docview.wss?uid=swg21993010.

quit

Description

Termine et ferme la session d'interface de ligne de commande AppScan Source. Emet une commande de déconnexion si vous êtes connecté.

Syntaxe

quit

Exemple

```
AllApplications>> Quit

>> Déconnecté.
>> Interface de ligne de commande Security AppScan Source existante...
```

record (rc)

Description

Active ou désactive l'enregistrement des commandes.

`Record` capture uniquement les commandes émises et leurs arguments, mais non pas leur sortie.

Conseil : Pour capturer les sorties, utilisez la commande `log`.

Syntaxe

record {*on* `fichier`|*off*}

- `on` ou `off` : requis pour activer ou désactiver la consignation.
- `fichier` : nom du fichier d'enregistrement. `Record On` requiert un argument `fichier` incluant un chemin d'accès valide. Si ce fichier n'existe pas encore, il est créé par l'interface de ligne de commande (CLI) d'AppScan Source. Si le fichier existe, les données de journal sont ajoutées à la fin du fichier.
- La commande `Record`, si elle n'est pas accompagnée d'arguments, signale si l'enregistrement dans un fichier est activé ou désactivé.

Exemple

- Pour consigner les informations dans un fichier nommé `MyCommands.txt`, entrez la commande suivante :

  ```
  AllApplications>> Record on C:\MyCommands.txt
  ```

- Pour désactiver la consignation des commandes, entrez la commande suivante :

  ```
  AllApplications>> RC off
  ```

refresh (rf)

Description

`Refresh` restaure le projet ou l'application en cours à partir du disque.

Utilisez la commande refresh lorsque vous voulez recharger un projet ou une application susceptible d'avoir été modifié.

Syntaxe

refresh

Exemple

Pour régénérer MyProject, entrez la commande suivante :

```
AllApplications\MyApplication\MyProject>> Refresh
```

register (reg)

Description

Enregistre des projets et des applications auprès de la base de données de AppScan Source.

Register permet d'enregistrer l'application ou le projet sur le noeud en cours auprès de AppScan Source. Leur enregistrement permet à AppScan Source d'avoir connaissance des applications et des projets. Les applications et les projets enregistrés peuvent être publiés (à l'aide de la commande publishassess (pa)).

Une fois qu'ils sont enregistrés, vous pouvez également annuler l'enregistrement des applications ou des projets à l'aide de la commande unregister (unreg).

Syntaxe

registre

Exemple

```
AllApplications>> cd pebble
AllApplications\pebble>> register
AllApplications\pebble>> L'enregistrement de 'pebble' a abouti.
```

removeassess (da)

Description

Supprime de la mémoire l'évaluation en cours ou celle sélectionnée.

Syntaxe

removeassess [*ID*]

ID : littéral facultatif. Identifie l'ID de l'évaluation. Si aucun ID n'est spécifié, la commande s'applique à l'évaluation la plus récente. Utilisez la commande listassess (la) pour trouver l'ID d'une évaluation.

Exemple

Pour supprimer une évaluation, déterminez d'abord son ID :

```
AllApplications\Applications>> ListAssess
542: putty (Application, Tue Apr 01 08:38:18 EDT 2008)
180: JavaAny (Application, Tue Apr 01 13:17:33 EDT 2008)
```

Supprimez ensuite l'évaluation portant l'ID 180 :

```
AllApplications\Applications>> RemoveAssess 180
```

report (rpt)

Description

`Report` génère un rapport AppScan Source du type spécifié, notamment des rapports sur les constatations et des rapports AppScan Source. Une licence valide pour AppScan Source for Automation est requise pour pouvoir utiliser cette commande.

Les formats de sortie de rapport disponibles sont HTML, PDF et zip.

Syntaxe

```
report "<report type>" <output format> <output location>
[<assessment id>] [-includeSrcBefore:<n>] [-includeSrcAfter:<n>]
[-includeTrace:<definitive|suspect|coverage>]
```

- `report type` : nom du rapport à générer, entre guillemets. Spécifiez l'une des options suivantes :
  - Rapport sur les constatations :
    - `Constatations par groupement`
    - `Constatations par API`
    - `Constatations par classification`
    - `Constatations`
    - `Activité DTS`
    - `Constatations par type`
    - `Constatations par énumération des faiblesses courantes (CWE)`
    - `Constatations par fichier`
  - Rapport AppScan Source :
    - `CWE SANS Top 25 2011`
    - `DISA Application Security and Development STIG V3R10`
    - `OWASP Mobile Top 10`
    - `OWASP Top 10 2013`
    - `PCI Data Security Standard V3.2`
    - `Profil de sécurité logicielle`
  - Rapport personnalisé, s'il en existe.

  Lorsque vous entrez le type de rapport entre guillemets, entrez-le exactement tel qu'il est spécifié dans la liste ci-dessus, par exemple `Constatations par classification` ou `Profil de sécurité logicielle`.
- `format de sortie` : spécifiez l'un des formats suivants pour ce rapport :
  - `html` : génère un rapport au format HTML et l'affiche en ligne.
  - `zip` : crée un fichier ZIP contenant tous les éléments du rapport HTML
  - Pour les rapports au format PDF, vous pouvez spécifier leur niveau de détail :
    - `pdf-summary` (pdf récapitulatif) : contient des comptages pour chaque groupe de rapport personnalisé
    - `pdf-detailed` (pdf-détaillé) : contient des comptages pour chaque API de chaque propriété d'une vulnérabilité
    - `pdf-comprehensive` (pdf-complet) : contient des tableaux couvrant chaque constatation pour chaque API

- `pdf-annotated` (pdf-annoté) : contient toutes les constatations, les notes éventuelles ajoutées aux constatations et les fragments de code pertinents
- `emplacement de sortie` : chemin de fichier pour l'écriture du rapport.

- `emplacement de sortie` : chemin absolu et nom du fichier dans lequel sauvegarder le rapport.
- `ID d'évaluation` : facultatif. ID de l'évaluation, que vous obtenez à l'aide de la commande `listassess (la)`. Si vous ne spécifiez pas cet ID, le rapport est généré à partir de l'analyse la plus récente.
- `-includeSrcBefore:<n>` : facultatif. Nombre de lignes du code source à inclure dans un rapport avant chaque constatation.
- `-includeSrcAfter:<n>` : facultatif. Nombre de lignes du code source à inclure dans un rapport après chaque constatation.
- `-includeTrace:<definitive|suspect|coverage>`: facultatif. Incluez les informations de trace dans le rapport pour les constatations définitives, suspectes ou les constatations de couverture (voir «Classifications», à la page 153 pour plus d'informations sur les classifications des constatations). Pour inclure les informations de trace d'autres classifications de constatation, indiquez cette option plusieurs fois. Par exemple, pour inclure les informations de trace des constatations définitives et suspectes, spécifiez `-includeTrace:definitive -includeTrace:suspect`.

Exemples

- Demande d'un rapport *Constatations par API* écrit au format HTML. Dans le rapport, incluez les informations de trace pour les constatations définitives :

```
AllApplications>> report "Findings by API" html
C:\reports\findings.html -includeTrace:definitive
```

- Pour générer un rapport *OWASP Top 10* AppScan Source au format PDF avec des détails complets à partir de l'évaluation existante 542 :

```
AllApplications>> report "OWASP Top 10 2013" pdf-comprehensive
/reports/webgoat_OWASP_13_comp.pdf 542
```

scan (sc)

Description

Analyse une application (ou toutes les applications), un projet ou un fichier. Une licence valide pour AppScan Source for Automation est requise pour pouvoir utiliser cette commande.

Important : Si vous travaillez avec un projet AppScan Source comportant des dépendances dans un environnement de développement (par exemple un projet IBM® MobileFirst Platform), veillez à créer le projet dans l'environnement de développement avant de l'importer. Après avoir importé le projet, si vous modifiez des fichiers qu'il contient, pensez à le régénérer dans l'environnement de développement avant l'analyse dans AppScan Source (si vous omettez cette étape, les modifications apportées aux fichiers ne seront pas prises en compte par AppScan Source).

Syntaxe

```
scan [path][config <proj_config>][-name <assessment_name>]
[-scanconfig <scan_configuration_name>]
```

- `path` : facultatif. Chemin d'accès complet et nom de fichier (`.ozasmt`) sous lesquels l'évaluation exportée sera sauvegardée.

Remarque :

- Si vous spécifiez un répertoire valide sans nom de fichier, un fichier d'évaluation (`.ozasmt`) est créé automatiquement en fonction du nom de l'application, du nom du projet et des configurations d'examen utilisés lors de la création de l'évaluation.
- Si vous spécifiez un répertoire valide avec un nom de fichier qui n'existe pas, un fichier d'évaluation est créé à cet emplacement avec le nom de fichier spécifié.
- Si vous spécifiez un fichier qui existe déjà, le fichier existant est remplacé.
- Si vous spécifiez un nom de fichier (`.ozasmt`) dans un répertoire qui n'existe pas, aucune évaluation n'est sauvegardée.

- `config <config_proj>` : facultatif. Cet argument est uniquement valide pour les évaluations de niveau projet. Si votre projet comporte un fichier de configuration, spécifiez-le à l'aide de cet argument.
- `-name <nom_évaluation>` : facultatif. Indiquez un nom pour l'évaluation. Ce nom est utilisé dans les produits client AppScan Source pour distinguer les évaluations (par exemple, dans AppScan Source for Analysis, le nom apparaît dans la colonne **Nom** de la vue Mes évaluations).
- `-scanconfig <nom_configuration_analyse>`: facultatif. Indiquez le nom de la configuration d'examen à utiliser pour l'examen. Si aucune configuration d'examen n'est spécifiée, c'est la configuration d'examen par défaut qui est utilisée.

Exemples

- Pour examiner les configurations par défauts des projets de toutes les applications, entrez la commande suivante :

  ```
  AllApplications>> Scan
  ```

  Les résultats apparaissent sous la forme suivante :

  ```
  New Scan started
    .
    .
  Preparing for Vulnerability Analysis...
  Performing Vulnerability Analysis...
  Generating Findings...
  Preparing project for scan...
    .
    .
  Scanned Project:
    Total files: 15
    Total findings: 167
    Total lines: 385
   vkloc: 0.44448395412925595
   v-Density: 22.446439683527426
  Scanned Application:
    Total files: 15
    Total findings: 167
    Total lines: 385
   vkloc: 0.44448395412925595
   v-Density: 22.446439683527426
  Scan completed:
    Total files: 15
    Total findings: 167
    Total lines: 385
   vkloc: 0.44448395412925595
   v-Density: 22.446439683527426
  Elapsed Time - 18 Seconds
  ```

```
New Scan started. Please wait...
Assessment complete
------------------
Total Call Sites: 75
Total Definitive Security Findings with High Severity: 25
Total Definitive Security Findings with Medium Severity: 37
Total Definitive Security Findings with Low Severity: 9
Total Suspect Security Findings with High Severity: 20
Total Suspect Security Findings with Medium Severity: 80
Total Suspect Security Findings with Low Severity: 60
Total Scan Coverage Findings with High Severity: 50
Total Scan Coverage Findings with Medium Severity: 33
Total Scan Coverage Findings with Low Severity: 17
Total Lines: 3000
...
```

- Pour examiner la configuration de débogage de `Prj1`, entrez la commande suivante :

```
AllApplications\Prj1>> SC config debug
```

script (scr)

Description

Exécute un script de commandes.

Le script peut contenir n'importe quelle commande interface de ligne de commande (CLI) d'AppScan Source valide et ses arguments. La commande `record (rc)` peut générer le fichier script.

Les scripts peuvent se référer à des variables d'environnement, à l'aide de la notation `${VAR}` (où `VAR` correspond au nom d'une variable d'environnement).

Lorsque la commande `script` est transmise à `AppScanSrcCli` depuis la ligne de commande, elle ferme automatiquement `AppScanSrcCli` à la fin du script.

Syntaxe

`script` `fichier_script`

`fichier_script` : chemin complet et nom du fichier script que vous souhaitez exécuter.

Commentaires dans les scripts

Une ligne précédée par un signe dièse (#) est un commentaire. L'interface de ligne de commande CLI ignore les commentaires lors d'une exécution en mode interactif ou depuis un fichier script.

L'exemple de script suivant comporte des commentaires :

```
login localhost User 123456
#navigate to my workspace
cd MyApplication
#scan my application
scan
logout
```

Exemples

Un utilisateur qui se connecte habituellement et utilise l'interface de ligne de commande CLI en mode interactif peut souhaiter enregistrer un script qu'il exécutera ensuite à de nombreuses reprises depuis la console. Vous pouvez créer un script d'une session entière à l'aide de la commande `record (rc)`, puis l'exécuter de façon répétée ou à intervalles planifiés. Vous pouvez utiliser un script pour automatiser une session complète en permettant à un seul fichier de commandes ou à un script de se connecter, de créer des projets, d'effectuer un examen, de copier des fichiers, etc.

- Pour exécuter un script nommé `myscript.txt`, entrez la commande suivante :

  ```
  AllApplications>> Script c:\myscript.txt
  ```

- L'exemple de script suivant se connecte, active la consignation au journal, crée une application, importe tous les projets `ppf` sous le répertoire indiqué, réalise une évaluation et se déconnecte :

  ```
  log on C:\mylogfile.log
  new MyApplication C:\myAppFolder
  cd MyApplication
  im c:\Projects\*.ppf
  examiner
  logout
  ```

  Notez que `logout` n'est pas nécessaire si le script s'exécute toujours depuis la ligne de commande `AppScanSrcCli`.

setaseinfo (sase)

Si votre serveur AppScan Enterprise Server a été installé avec l'option AppScan Enterprise Console, vous pouvez publier des évaluations sur cette console. Le composant Enterprise Console offre divers outils de gestion des évaluations, tels que des fonctions de génération de rapports, de gestion de problème et d'analyse de tendance, ainsi que des tableaux de bord.

Description

Spécifiez les paramètres Enterprise Console.

Si vous avez l'intention de publier des évaluations depuis AppScan Source vers Enterprise Console, utilisez la commande `setaseinfo` pour indiquer les paramètres de connexion à Enterprise Console.

Syntaxe

```
setaseinfo --userid|-u <id utilisateur>
--password|-p <mot de passe>
--url|-l <url>
```

- `userid` : obligatoire.
  - Si vous utilisez un ID utilisateur et un mot de passe comme méthode d'authentification AppScan Enterprise Server, entrez votre ID utilisateur.
  - Si votre serveur AppScan Enterprise Server est configuré pour utiliser l'authentification Windows, entrez le nom de domaine et d'utilisateur que vous utilisez pour vous connecter au serveur Enterprise Console (en les séparant par un signe \. Par exemple, `my_domain\my_username`).
  - Si votre serveur AppScan Enterprise Server est configuré avec LDAP, entrez le nom d'utilisateur que vous utilisez pour vous connecter au serveur Enterprise Console.

– Sous Windows, si votre serveur AppScan Enterprise Server est activé pour l'authentification CAC (Common Access Card), entrez `common_name<cn_certificate>`, où `common_name` est votre nom usuel CAC et `cn_certificate` est le nom usuel de l'émetteur de votre certificat. Si votre nom usuel (`common_name`) ou votre certificat CN (`cn_certificate`) comporte des espaces, placez-les entre guillemets (`"common_name<cn_certificate>"`).

Au minimum, vous devez être un utilisateur QuickScan. Si vous êtes connecté à un AppScan Enterprise Server de version antérieure à la version 9.0.3, vous disposez normalement de votre propre dossier utilisateur sur le Enterprise Server.

- `password` : Requis si vous utilisez un ID utilisateur et un mot de passe comme méthode d'authentification AppScan Enterprise Server. Spécifiez le mot de passe associé à votre ID utilisateur Enterprise Server.

  Remarque : Cette option n'est pas utilisée lorsque votre serveur AppScan Enterprise Server est activé pour l'authentification CAC.

- `url` : Spécifiez l'URL de votre instance AppScan Enterprise Server. Le format de cette URL est le suivant : `http(s)://<nom_d'hôte>:<port>/ase`, où `<nom_d'hôte>` correspond au nom de la machine sur laquelle AppScan Enterprise Server est installé, et `<port>`, au port sur lequel le serveur s'exécute. Par exemple, `https://myhost.mydomain.ibm.com:9443/ase`.

  Remarque : Cet argument est facultatif si l'adresse URL de Enterprise Console a déjà été définie.

Remarque :

- Pour pouvoir définir l'`url`, vous devez être connecté à l'interface de ligne de commande (CLI) d'AppScan Source avec l'autorisation `Gérer les paramètres AppScan Enterprise`. Pour plus d'informations sur les comptes utilisateur et les autorisations, reportez-vous à la section *Administration de AppScan Source* dans le manuel *IBM Security AppScan Source - Guide d'installation et d'administration*.
- L'`ID utilisateur` et le `mot de passe` sont stockés sur le poste sur lequel le client AppScan Source s'exécute (par exemple, AppScan Source for Analysis), tandis que l'`url` est stockée sur Enterprise Server (qui peut se trouver sur un poste distant). Vous ne pouvez pas accéder aux informations d'`id utilisateur` et de `mot de passe` à partir du poste distant (par exemple, en émettant la commande `getaseinfo`).

Exemple

```
AllApplications>> setaseinfo --userid nom_utilisateur
 --password mot_passe --url https://serveur_ase/ase
Les informations d'AppScan Enterprise Integration ont été configurées.
AllApplications>>
```

setcurrentobject (set, cd)

Description

Utilisez la commande `setcurrentobject` pour naviguer dans l'arborescence.

Cette commande définit l'objet actuellement sélectionné dans l'arborescence. L'objet doit être un enfant valide de l'objet en cours. Pour afficher les enfants de l'objet en cours, utilisez la commande `list (ls, dir)`. Sélectionnez l'objet d'après son nom ou son ID.

Syntaxe

setcurrentobject {nom|*ID* ID_objet}

- nom : nom de l'enfant à définir. Requis lors d'une navigation dans l'arborescence d'après le nom de l'objet.
- *ID* : littéral. Sélection d'après l'ID de l'objet.
- ID_objet : ID numérique de l'enfant à définir. Requis en cas de sélection d'après l'ID.

Exemples

- Pour naviguer depuis la racine jusqu'à l'application portant l'ID 1, entrez la commande suivante :

```
AllApplications>> SetCurrentObject applications
AllApplications>> CD id 1
AllApplications\Applications>>
```

- Pour remonter d'un niveau, entrez la commande suivante :

```
AllApplications\Applications>> CD ..
```

- Pour revenir au sommet (racine) de l'arborescence des objets, entrez la commande suivante :

```
AllApplications\Applications>> Set /
```

ou

```
AllApplications\Applications>> Set \
```

setvar (sv)

Description

Crée une nouvelle variable ou modifie une variable existante.

Syntaxe

setvar <nom> <chemin>

- nom : nom de la variable. Si cette variable existe déjà, l'interface de ligne de commande (CLI) d'AppScan Source met à jour son chemin d'accès. Si la variable n'existe pas, l'interface CLI la crée avec la valeur du chemin nommé.
- chemin : chemin existant pour la variable. Si ce chemin n'existe pas, la variable n'est pas créée.

Exemples

- Pour créer une variable AppScan Source nommée Myvar avec le chemin C:\Myapps, entrez la commande suivante :

```
AllApplications>> SetVar Myvar C:\Myapp
AllApplications>> La nouvelle variable '%Myvar%' a été créée
```

- Pour créer une variable AppScan Source nommée SRC_ROOT d'après la valeur de la variable d'environnement système SRC_ROOT, entrez la commande suivante :

```
# connexion
login localhost admin
# création de la variable SRC_ROOT avec la valeur
# de la variable d'environnement SRC_ROOT
setvar SRC_ROOT ${SRC_ROOT}
```

unregister (unreg)

Description

Cette commande permet d'annuler l'enregistrement d'une application ou d'un projet précédemment enregistré dans le noeud en cours.

Après cette opération, l'application ou le projet ne peuvent plus être publiés. Pour plus d'informations, voir la commande `register (reg)`.

Syntaxe

```
unregister
```

Exemple

```
AllApplications\Myapp>> Unregister
AllApplications\Myapp>> L'annulation de l'enregistrement de 'Myapp' a abouti.
```

Exécution d'évaluations automatisées

L'interface de ligne de commande (CLI) d'AppScan Source vous permet d'importer automatiquement un fichier de projet AppScan Source (`.ppf`) et d'examiner votre code source. Depuis la ligne de commande, vous pouvez exécuter un script, comme dans l'exemple `Run_Assessments.txt` ci-dessous.

```
AppScanSrcCli scr c:\<install_dir>\bin\Run_Assessments.txt
```

Exemple : Run_Assessments.txt

```
# Log in.
Login <nom_hôte> <nom_utilisateur> <nom_utilisateur>
# Turn on logging.
log on c:\myLogFile.log
# Create a new application named "testit."
new testit c:\AppTest
# Navigate to the newly created application.
cd testit
# Import the Project files (.ppfs) under c:\projects\joans.
im c:\projects\joans\*.ppf
# Refresh the Project.
refresh

# Run an assessment.
scan
# Register the assessment
register
# Publish the assessment
publishassess
# Log out and end the CLI session.
quit
```

Sortie

```
Logging to 'c:\mylogfile.log'...

AllApplications>> new testit c:\AppTest
AllApplications>> cd testit
AllApplications\testit>> import c:\TestApps\testproj\*.ppf
AllApplications\testit>> refresh
AllApplications\testit>> ls

214: testproj (Project [local])

AllApplications\testit>> la
```

```
testit has no current assessments.

scan
New Scan started at 15:41:55
Scanning Project testproj (1 of 1)
Preparing project for scan...
.
.
.

Searching File C:\TestApps\\testproj\src\se\bluefish\blueblog\metarepository\Meta
Category.java (21 of 33)
Searching File C:\TestApps\testproj\src\se\bluefish\blueblog\metarepository\Meta
Repository.java (22 of 33)

-------------------
Total Call Sites: 348
Total Definitive Security Findings with High Severity: 5
Total Definitive Security Findings with Medium Severity: 1
Total Definitive Security Findings with Low Severity: 4
Total Suspect Security Findings with High Severity: 0
Total Suspect Security Findings with Medium Severity: 8
Total Suspect Security Findings with Low Severity: 0
Total Scan Coverage Findings with High Severity: 16
Total Scan Coverage Findings with Medium Severity: 27
Total Scan Coverage Findings with Low Severity: 16
Total Lines: 7386
Max V-Density: 732.2772813430815
Max V/kloc: 10.42512862171676
V-Density: 732.2772813430815
V/kloc: 10.42512862171676

AllApplications\testit>> register
AllApplications\testit>> 'testit' registered successfully.
AllApplications\testit>> pa

Assessment Successfully Published.

AllApplications\testit>> la

AllApplications\testit>> 27001: testit (Application, Fri Mar 14 15:41:55 EDT 2008)

AllApplications\testit>>
```

Chapitre 3. Outil de génération Ounce/Ant

Cette section décrit comment utiliser Ounce/Ant, qui est un utilitaire de génération AppScan Source intégrant AppScan Source et Apache Ant. L'intégration d'Ounce/Ant avec votre environnement Ant facilite l'automatisation des générations et des évaluations du code.

Ounce/Ant est un outil de génération AppScan Source qui s'intègre dans votre environnement de génération Apache Ant. Au lieu de créer et de configurer manuellement des fichiers d'application et de projet AppScan Source dans AppScan Source for Analysis, Ounce/Ant génère automatiquement des projets et des applications AppScan Source contenant toutes les informations requises, telles que les fichiers source et les chemins de classes. AppScan Source peut ensuite examiner les fichiers de projet et d'application.

Apache Ant version 1.7 ou ultérieure est requise pour une utilisation avec Ounce/Ant.

Intégration d'Ounce/Ant et d'Apache/Ant

Les étapes décrites dans cette rubrique de tâche sont nécessaires pour intégrer Ounce/Ant dans l'environnement de génération Apache/Ant.

Procédure

1. Copiez le fichier `ounceant.jar` et, si vous le souhaitez, le fichier `ant-contrib-1.0b3.jar` dans le répertoire `ant lib`.

   L'installation de AppScan Source place les fichiers `ounceant.jar` et `ant-contrib-1.0b3.jar` dans `<install_dir>\lib` (où `<rép_install>` représente l'emplacement de votre installation AppScan Source).
2. Facultatif : Redéfinissez la propriété `build.compiler`.

   Voir «Création de projets», à la page 65 pour plus d'informations sur la redéfinition de `build.compiler`.)

   Redéfinissez cette propriété à l'aide d'une des méthodes suivantes :

   - Utilisez une balise de propriété dans le fichier `build.xml`.

     ```
     <property name="build.compiler"
     value="com.ouncelabs.ounceant.OunceCompilerAdapter"/>
     ```

   - Spécifiez la valeur de `build.compiler` dans la ligne de commande lors de l'appel Ant à l'aide de l'option `-D`.

     ```
     ant -Dbuild.compiler=
     com.ouncelabs.ounceant.OunceCompilerAdapter
     ```

   - Incluez la valeur de `build.compiler` dans un fichier texte et indiquez à Ant de charger les propriétés de ce fichier à l'aide de l'option `properties` comme décrit dans la documentation Ant.
3. Créez `taskdefs`.

   Pour utiliser des tâches Ounce/Ant, vous devez référencer `ounceant.jar` dans une tâche `taskdef`. Par exemple,

   ```
   <taskdef resource="com/ouncelabs/ounceant/task/ounce.xml"
   classpath="ounceant.jar"/>
   ```

   Pour utiliser la tâche `var`, `ant-contrib-1.0b3.jar` doit référencer la tâche `taskdef` comme suit :

```
<taskdef resource="net/sf/antcontrib/antlib.xml"/>
```

Propriétés Ounce/Ant

Le tableau de cette rubrique décrit les propriétés Ounce/Ant facultatives.

Propriété	Valeurs valides / Valeur par défaut	Description
`ounce.default.configuration_name`	`<config_name>`	Nom de configuration créé dans le projet AppScan Source. S'il n'est pas défini, le nom Configuration 1 est utilisé.
`ounce.build.compiler`	nom du compilateur	Nom d'exécutable du compilateur à utiliser lorsque la propriété Ant `build.compiler` est configurée pour utiliser `OunceCompilerAdapter`. S'il n'a pas été défini, `javac` est utilisé.
`ounce.project_name`	nom	Nom du projet. S'il n'a pas été défini, Ounce/Ant en génère un. Pour plus d'informations, voir «Désignation de projets» , à la page 67.
`ounce.project_dir` (Requis pour définir `ounce.project_name`)	`<project_dir>`	Répertoire de travail du projet S'il n'a pas été défini, le répertoire actuel des fichiers de génération est utilisé.
`ounce.application_name`	`<app_name>`	Nom de l'application. Voir «Création et désignation d'applications», à la page 67.
`ounce.application_dir` (Requis pour définir `ounce.application_name`)	`<app_dir>`	Répertoire de travail de l'application. Vous devez définir `ounce.application_name` et `ounce.application_dir` pour ajouter un projet à une application.
`ounce.projects.store_full_paths` (facultatif)	`true` ou `false`	Stocke les chemins absolus créés dans les applications et les projets. S'il n'est pas défini, des chemins d'accès relatifs sont utilisés.

Configuration des propriétés

Vous pouvez utiliser la tâche `var` pour configurer des propriétés autant de fois que nécessaire. Lorsque `ounceCreateProject` est appelé, explicitement ou implicitement, les valeurs de propriétés en cours sont utilisées. Vous pouvez configurer des propriétés à l'aide d'une des méthodes évoquées dans cette rubrique de tâche.

Procédure

- Utilisez un attribut sur une tâche Ounce/Ant si l'attribut existe.
- Utilisez une tâche `var` avant la tâche AppScan Source appropriée. La portée des propriétés définies avec `var` n'est conservée que pour le fichier dans lequel elles sont définies. Pour utiliser la tâche `var`, la bibliothèque de contribution Ant doit figurer dans le répertoire Ant `lib` et être référencée par la tâche `taskdef`.

  ```
  <taskdef resource="net/sf/antcontrib/antlib.xml"/>
  ```
- Utilisez l'option `-D` sur la ligne de commande Ant.
- Placez la propriété dans un fichier de propriétés de génération (habituellement nommé `build.properties`).

Création de projets

La tâche `ounceCreateProject` génère des projets AppScan Source.

Vous pouvez utiliser `ounceCreateProject` explicitement ou implicitement :

- Vous pouvez appeler explicitement `ounceCreateProject` dans un fichier de génération Ant.
- Vous pouvez appeler implicitement `ounceCreateProject` chaque fois que la tâche `javac` est appelée en remplaçant la propriété `build.compiler`.

Les projets peuvent être regroupés en applications. Vous pouvez utiliser l'attribut facultatif d'application ou les propriétés Ounce/Ant appropriées pour placer le projet créé dans une application.

Les projets peuvent également être augmentés à l'aide d'Ounce/Ant. Les attributs de projet de plusieurs projets peuvent être fusionnés dans un même projet AppScan Source. Si plusieurs `javac` ou `ounceCreateProject` doivent augmenter le même projet, conservez le même nom et répertoire.

Remarque : Si vous remplacez `build.compiler` pour créer des projets, exécutez d'abord `ant clean`.

Reportez-vous à la rubrique «Exemple de tâche ounceCreateProject», à la page 66 pour déterminer l'utilisation des propriétés et de leurs attributs imbriqués.

ounceCreateProject

`ounceCreateProject` accepte les paramètres et les éléments imbriqués suivants :

Paramètres spécifiés en tant qu'attributs	Description
`name`	Nom du projet
`workingDir`	Répertoire de travail pour le projet.
`classpath`	Chemin de classes pour le projet.

Paramètres spécifiés en tant qu'attributs	Description
sourcepath	Chemin source pour les dépendances source de projet. Ces fichiers ne sont pas analysés pour détecter des vulnérabilités.
jdkName	Nom du kit JDK AppScan Source à utiliser pour l'examen d'un projet. (Doit être créé dans AppScan Source for Analysis)
appName	Application contenant ce projet (facultatif).
appDir	Répertoire d'application (facultatif)

Paramètres spécifiés en tant qu'éléments imbriqués	Description
ounceSourceRoot	Spécifiez les racines sources du projet
ounceWeb	Spécifiez le contenu du projet Web
ounceExclude	Permet l'exclusion de certains fichiers de l'ensemble de fichiers spécifié dans l'élément parent.

Exemple de tâche ounceCreateProject

```
<ounceCreateProject
 name="myProjectName"
 workingDir="${sandbox}/myProject"
 classpath="${my.class.path}"
 sourcepath="${my.source.path}"
 jdkName="jdk15"
 appName="myApp"
 appDir="${sandbox}>
<ounceSourceRoot dir="src"/>
<ounceExclude dir="src/test"/>
<ounceExclude file="src/mydir/Bad.java"/>
<ounceSourceRoot dir="src2"/>
<ounceSourceRoot file="src3/mydir.java"/>
<ounceWeb webContextRoot="web/myProject.war"/>
<ounceExclude dir="web/test"/>
<ounceExclude file="web/partial.jsp"/>
</ounceCreateProject>
```

ounceSourceRoot

ounceSourceRoot accepte les paramètres suivants spécifiés en tant qu'attributs :

Paramètres spécifiés en tant qu'attributs	Description
dir	Chemin d'une racine source Java™ valide
file	Chemin d'un fichier individuel

ounceWeb

ounceWeb accepte les paramètres suivants spécifiés en tant qu'attributs.

Paramètres spécifiés en tant qu'attributs	Description
webContextRoot	Chemin d'une racine Web valide ou d'un fichier war.

ounceExclude

ounceExclude accepte les paramètres suivants spécifiés en tant qu'attributs :

Paramètres spécifiés en tant qu'attributs	Description
dir	Chemin d'un répertoire à exclure
file	Chemin d'un fichier individuel à exclure

Désignation de projets

Vous pouvez nommer un projet à l'aide de la convention de dénomination par défaut ou en spécifiant un nom de projet manuellement.

Désignation par défaut de projet

Le schéma de dénomination par défaut d'Ounce/Ant permet de générer des noms de projets uniques.

<répertoire>_<nom_cible>[_<numéro>]

- <répertoire> : nom du répertoire de projet (n'inclut pas le chemin complet)
- <nom_cible> : nom de la cible en cours
- <numéro> : entier commençant à 1 et incrémenté comme nécessaire pour garantir qu'il soit unique. Notez que le <numéro> apparaît uniquement en cas de conflit.

Exemple

En utilisant les paramètres suivants, le nom de projet devient test_compile. Si deux projets sont créés sur la cible de compilation, le second projet reçoit le nom test_compile_1.

```
<répertoire_travail> = C:\mydir\test
<répertoire> = test
<nom_cible> = compile
```

Désignation manuelle de projets

Si vous créez un projet à l'aide d'ounceCreateProject, vous pouvez utiliser l'attribut name afin de spécifier un nom de projet. Si l'attribut name n'est pas spécifié ou si vous créez des projets à l'aide de javac, la tâche AppScan Source examine la propriété ounce.project_name et, si celle-ci a été définie, utilise sa valeur comme nom de projet. Par conséquent, la propriété ounce.project_name peut être définie à l'aide de var avant chaque appel de la tâche javac en indiquant un nom unique pour chaque projet créé.

Création et désignation d'applications

Ounce/Ant peut créer des applications AppScan Source lors de son fonctionnement et regrouper les projets AppScan Source créés dans les applications.

Ces applications sont créées à l'aide de deux propriétés Ant :

- ounce.application_name
- ounce.application_dir

Important : Pour créer l'application, vous devez soit définir les deux propriétés, soit définir les attributs `appName` et `AppDir` de la tâche `ounceCreateProject`.

Si ces propriétés sont définies lors de la création du projet, celui-ci est ajouté à l'application portant le nom spécifié par `ounce.application_name` et au répertoire de travail spécifié par `ounce.application_dir`.

Vous pouvez définir ces propriétés à plusieurs reprises à l'aide de la tâche `var` afin de regrouper les projets dans les applications voulues. Un nom d'application approprié reflète la combinaison de projets dans une seule application.

Conseil : Si vous utilisez `ounceCreateProject`, vous pouvez également utiliser les attributs `appName` et `appDir` afin de spécifier l'application dans laquelle inclure le projet.

Intégration de génération

La tâche `ounceCli` permet d'accéder à l'interface de ligne de commande (CLI) d'AppScan Source pour effectuer un examen pendant une génération. La tâche `ounceCli` peut appeler `cli` depuis Ant.

`ounceCli` requiert que les paramètres suivants soient spécifiés en tant qu'attributs :

- `dir` : emplacement du répertoire d'installation de AppScan Source
- `script` : emplacement du fichier script `cli` à exécuter
- `output` : emplacement du fichier de sortie `cli`

Exemple

```
<ounceCli
dir="${installDir}"
 script="${scripts}/cli_script.txt"
 output="log.txt"/>
```

Chapitre 4. API d'accès aux données AppScan Source

L'API d'accès aux données fournit l'accès aux résultats d'évaluation générés par AppScan Source, y compris aux constatations et aux détails des constatations. Elle donne également accès aux métriques d'évaluation telles que la date et l'heure de l'analyse, le nombre de lignes de code, la densité V et le nombre de constatations.

L'API d'accès aux données est incluse dans l'installation des composants AppScan Source suivants :

- AppScan Source for Analysis
- interface de ligne de commande (CLI) d'AppScan Source

L'API d'accès aux données est installée sous `<install_dir>\sdk\ouncesdk.jar` sous Windows et sous `<install_dir>/sdk/ouncesdk.jar` sous Linux (où `<rép_install>` représente l'emplacement de votre installation AppScan Source).

L'API d'accès aux données requiert le kit JDK 1.5, ou ultérieur.

Sous Windows : Lors de l'exécution d'un programme qui utilise le SDK, indiquez les arguments JVM (Java Virtual Machine) suivants dans la ligne de commande :

```
java -classpath
<install_dir>\lib\avalon-framework-4.1.5.jar;
<install_dir>\lib\commons-lang3-3.3.2.jar;
<install_dir>\lib\icu4j-52_1.jar;
<install_dir>\lib\jacorb.jar;
<install_dir>\lib\log4j-1.2.8.jar;
<install_dir>\lib\logkit-1.2.jar;
<install_dir>\sdk\ouncesdk.jar;
<install_dir>\lib\xml-apis.jar;
<install_dir>\lib\saxon9.jar
... com.company.product.ClassName
```

Sous Linux : lors de l'exécution d'un programme qui utilise le SDK, spécifiez ces arguments Java Virtual Machine (JVM) dans la ligne de commande :

```
java -classpath
<install_dir>/lib/avalon-framework-4.1.5.jar:
<install_dir>/lib/commons-lang3-3.3.2.jar:
<install_dir>/lib/icu4j-52_1.jar:
<install_dir>/lib/jacorb.jar:
<install_dir>/lib/log4j-1.2.8.jar:
<install_dir>/lib/logkit-1.2.jar:
<install_dir>/sdk/ouncesdk.jar:
<install_dir>/lib/xml-apis.jar:
<install_dir>/lib/saxon9.jar
... com.company.product.ClassName
```

Modèle d'objet d'API d'accès aux données

Diagramme 1 - Diagramme UML (langage de modélisation unifié) détaillant le modèle d'objet des évaluations

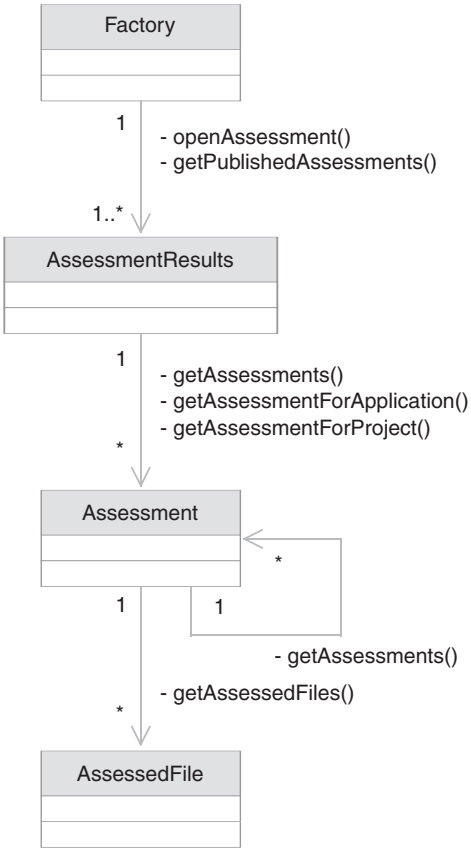


Diagramme 2 - Diagramme UML (langage de modélisation unifié) détaillant le modèle d'objet des constatations

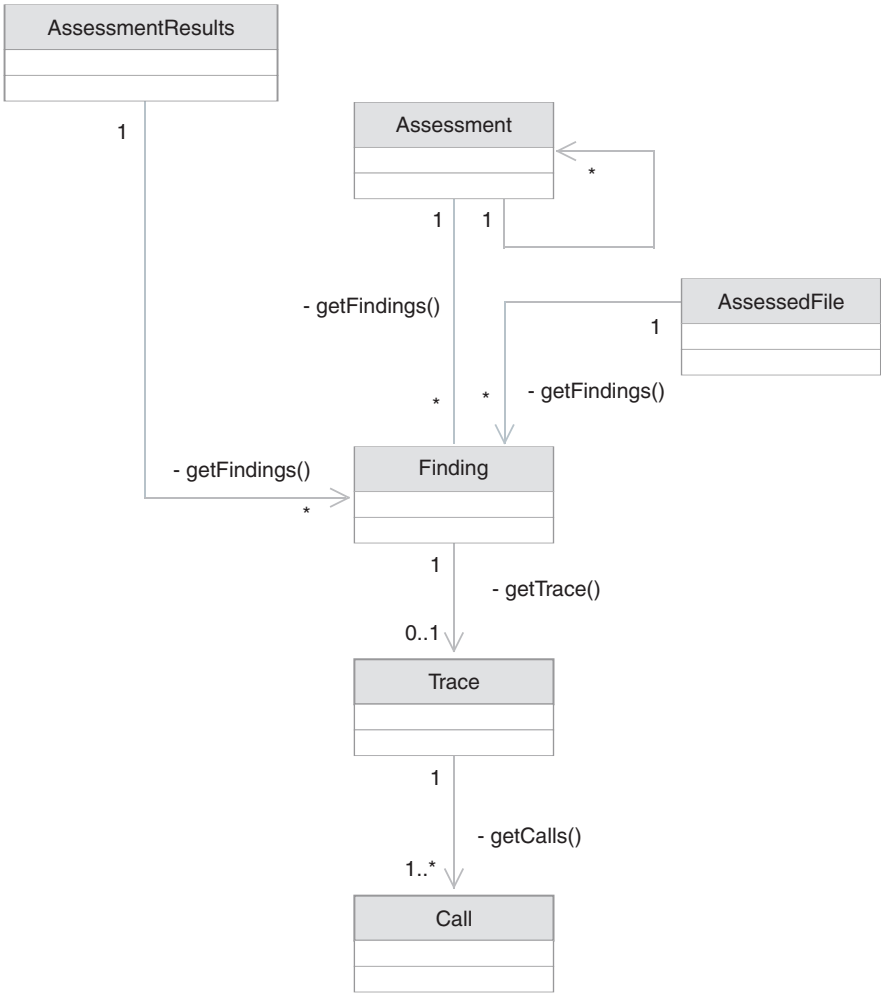


Diagramme 1 - Diagramme UML (langage de modélisation unifié) détaillant le modèle d'objet des évaluations

La classe `Factory` se trouve au sommet du diagramme. La classe `Factory` apparaît comme étant liée à la classe `AssessmentResults` via deux méthodes dans la classe `Factory` : `openAssessment()` et `getPublishedAssessments()`. La relation entre ces deux classes est une classe `Factory` associée à une ou plusieurs classes `AssessmentResults`.

Sous la classe `Factory`, le diagramme représente la classe `AssessmentResults`. La classe `AssessmentResults` apparaît comme étant liée à la classe `Assessment` via trois méthodes : `getAssessments()`, `getAssessmentForApplication()` et `getAssessmentForProject()`. La relation entre ces deux classes est une classe `AssessmentResults` associée à une ou plusieurs classes `Assessment`.

Sous la classe `AssessmentResults`, le diagramme représente la classe `Assessment`. Une classe `Assessment` est associée à zéro ou plusieurs classes `AssessedFile` via la méthode `getAssessedFiles()`. La classe `Assessment` représente également une relation avec elle-même via la méthode `getAssessments()`. Cette relation est établie entre une classe `Assessment` et zéro ou plusieurs classes `Assessment` enfant.

Sous la classe Assessment, le diagramme représente la classe AssessedFile. Une classe Assessment est associée à zéro ou plusieurs classes AssessedFile via la méthode getAssessedFiles().

Diagramme 2 - Diagramme UML (langage de modélisation unifié) détaillant le modèle d'objet des constatations

Trois classes se trouvent au sommet du diagramme :AssessmentResults, Assessment et AssessFile. Chacune de ces trois classes sont représentées comme étant liées à la classe Finding via une méthode getFindings(). La relation est établie entre une classe et zéro ou plusieurs classes Finding. La classe Assessment représente également une relation avec elle-même établie entre une classe Assessment et zéro ou plusieurs classes Assessment enfant.

Sous la classe Finding, le diagramme représente la classe Trace. Une classe Finding est associée à zéro ou plusieurs classes Trace via la méthode getTrace().

Sous la classe Trace, le diagramme représente la classe Call. Une classe Trace est associée à une ou plusieurs classes Call via la méthode getCalls().

Utilisation de l'API d'accès aux données

Des exemples complets figurent dans les fichiers SamplePublished.java et SampleSdk.java inclus dans <install_dir>\sdk\sample\com\ouncelabs\sdk\sample (où <rép_install> représente l'emplacement de votre installation AppScan Source).

Classes et méthodes de l'API d'accès aux données

Cette section décrit en détail les classes et les méthodes d'accès aux données AppScan Source. Les classes sont les suivantes :

- com.ouncelabs.sdk.Factory
- com.ouncelabs.sdk.assessment.AssessedFile
- com.ouncelabs.sdk.assessment.Assessment
- com.ouncelabs.sdk.assessment.AssessmentDiff
- com.ouncelabs.sdk.assessment.AssessmentResults
- com.ouncelabs.sdk.assessment.Finding
- com.ouncelabs.sdk.assessment.Trace
- com.ouncelabs.sdk.assessment.Call
- com.ouncelabs.sdk.assessment.SeverityType
- com.ouncelabs.sdk.assessment.ClassificationType
- com.ouncelabs.sdk.assessment.AssessmentFilter
- com.ouncelabs.sdk.assessment.OunceException

AssessedFile

La classe com.ouncelabs.sdk.assessment.AssessedFile représente une évaluation d'un fichier individuel. Elle permet l'accès aux données d'évaluation concernant un fichier, par exemple ses constatations et statistiques.

AssessedFile.getFindings

Signature

```
public Finding[] getFindings()
```

Description

Permet l'accès à toutes les constatations relatives à cet élément `AssessedFile`.

Retour

Tableau des objets constatations pour le fichier évalué.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les constatations.

`AssessedFile.getStats`

Signature

```
public AssessmentStats getStats()
```

Description

Fournit un accès aux statistiques agrégées pour ce fichier, telles que le nombre total de constatations, le nombre total de lignes, etc.

Retour

Objet `AssessmentStats` contenant les statistiques pour cette évaluation.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les statistiques.

`AssessedFile.getFilename`

Signature

```
public String getFilename()
```

Description

Permet l'accès au chemin absolu du fichier représenté par cet élément `AssessedFile`.

Retour

Chaîne contenant le chemin absolu du fichier représenté par cet élément `AssessedFile`.

`Assessment`

La classe `com.ouncelabs.sdk.assessment.Assessment` représente une évaluation d'une application ou d'un projet.

`Assessment.getFindings`

Signature

```
public Finding[] getFindings()
```

Description

Fournit un accès à toutes les constatations de cette évaluation.

Retour

Tableau d'objets constatation pour cette évaluation.

Assessment.getStats

Signature

```
public AssessmentStats getStats()
```

Description

Liste complète de statistiques comprenant le nombre total de fichiers analysés, le nombre total de vulnérabilités détectées, l'heure de l'exmen et la densité des vulnérabilités.

Retour

Objet `AssessmentStats` contenant les statistiques pour cette évaluation.

Assessment.getAssessments

Signature

```
public Assessment[] getAssessments()
```

Description

Renvoie les objets `Assessment` (Evaluation) enfant pour cette évaluation. Vu que les évaluations de projets n'ont pas d'enfants, les objets enfant ne sont renvoyés que pour les évaluations d'applications.

Retour

- Pour les évaluations d'application, renvoie un tableau d'objets `Evaluation` comportant un objet pour chaque projet analysé dans l'application.
- Pour les évaluations de projet, renvoie un tableau vide.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les évaluations.

Assessment.getFiles

Signature

```
public AssessedFile[] getFiles()
```

Description

Fournit un accès à chaque fichier faisant partie de cette évaluation.

Retour

Tableau d'objets `AssessedFile`. Chaque objet représente un fichier faisant partie de cette évaluation.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les fichiers évalués.

Assessment.getFileByPath

Signature

```
public AssessedFile getFileByPath(filePath)
```

Description

Fournit un accès a un fichier évalué d'après son chemin.

Retour

Objet `AssessmentFile` correspondant au chemin de fichier spécifié ou la valeur Null si ce chemin ne correspond à aucun fichier évalué.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les fichiers évalués.

Assessment.getAssesseeName

Signature

```
public String getAssesseeName()
```

Description

Fournit le nom de l'application ou du projet depuis lequel cette évaluation a été générée.

Retour

Nom d'une application ou d'un projet.

AssessmentDiff

La classe `com.ouncelabs.sdk.assessment.AssessmentDiff` contient la différence entre deux évaluations, en fournissant le delta entre les deux.

AssessmentDiff.getCommonFindings

Signature

```
public Finding[] getCommonFindings()
```

Description

Permet d'obtenir les constatations communes aux deux évaluations.

AssessmentDiff.getLostFindings

Signature

```
public Finding[] getLostFindings()
```

Description

Permet d'obtenir les constatations qui étaient présentes dans l'évaluation la plus ancienne, mais pas dans la plus récente.

AssessmentDiff.getNewFindings

Signature

```
public Finding[] getNewFindings()
```

Description

Permet d'obtenir les constatations qui étaient présentes dans l'évaluation la plus récente, mais pas dans la plus ancienne.

AssessmentFilter

Utilisez la classe `com.ouncelabs.sdk.assessment.AssessmentFilter` pour spécifier des critères de filtrage lors de l'extraction d'évaluations publiées.

AssessmentFilter.AssessmentFilter

Signature

```
public AssessmentFilter(int maxResults)
```

Description

Constructeur spécifiant uniquement le nombre maximal d'évaluations publiées à extraire. Vous pouvez spécifier des options de filtrage supplémentaires en appelant les méthodes Set de la classe (par exemple, `setUserName()` pour filtrer par utilisateur publié).

Paramètres

- `maxResults` : nombre maximal de résultats à renvoyer.

AssessmentFilter.AssessmentFilter

Signature

```
public AssessmentFilter(String appName,
String userName,
Double dateProximityDuration,
int DateProximityUnit,
Long dateRangeStart, Long dateRangeEnd,
int maxResults)
```

Description

Constructeur acceptant toutes les options de critères comme arguments.

Paramètres

- `appName` : nom de l'application. (Null si le filtrage n'est pas effectué d'après `appName`).
- `userName` : utilisateur ayant publié l'application. (Null si le filtrage n'est pas effectué d'après le nom d'utilisateur).

- `dateProximityDuration` : lorsqu'il est couplé avec le paramètre `dateProximityUnit`, indique le nombre d'unités (jours, semaines, etc.) à filtrer à partir de la date actuelle. (Null si le filtrage n'est pas effectué par proximité de date).
- `dateProximityUnit` : lorsqu'il est couplé avec le paramètre `dateProximityDuration`, indique l'unité (jours, semaines, etc.) d'après laquelle effectuer le comptage. Requis si `dateProximityDuration` a été spécifié. Voir «`DateProximityUnit.value`», à la page 82 pour déterminer les unités valides.
- `dateRangeStart` : début de la plage de dates. (Null si le filtrage n'est pas effectué par plage de dates).
- `dateRangeEnd` : fin de la plage de dates (Null si le filtrage n'est pas effectué par plage de dates).
- `maxResults` : nombre maximal de résultats à renvoyer.

AssessmentResults

La classe `com.ouncelabs.sdk.assessment.AssessmentResults` représente l'évaluation complète.

AssessmentResults.getFindings

Signature

```
public Finding[] getFindings()
```

Description

Permet l'accès à toutes les constatations relatives à cet élément `AssessmentResults`.

Retour

Tableau d'objets constatation pour cette évaluation.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les constatations.

AssessmentResults.getAssessments

Signature

```
public Assessment[] getAssessments()
```

Description

Fournit un accès aux données d'évaluation pour chaque application évaluée dans le cadre de cet objet `AssessmentResults`.

Retour

Tableau d'objets Evaluation comportant un objet pour chaque application faisant partie des `AssessmentResults`.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les évaluations.

AssessmentResults.getStats

Signature

```
public AssessmentStats getStats()
```

Description

Fournit un accès aux statistiques agrégées pour cette évaluation, telles que le nombre total de fichiers évalués, le nombre total de lignes, etc.

Retour

Objet `AssessmentStats` contenant les statistiques pour cette évaluation.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les statistiques de l'évaluation.

AssessmentResults.getAssessmentForApplication

Signature

```
public Assessment[] getAssessmentForApplication(String applicationName)
```

Description

Fournit un accès aux données d'évaluation pour l'application spécifiée.

Paramètres

`applicationName` : nom de l'application.

Retour

Tableau d'objets Evaluation comportant un objet pour chaque application correspondant au nom spécifié.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les évaluations.

AssessmentResults.getAssessmentForProject

Signature

```
public Assessment[]
getAssessmentForProject(String projectName, String applicationName)
```

Description

Fournit un accès aux données d'évaluation pour le projet indiqué dans l'application spécifiée.

Paramètres

`projectName` : nom du projet.

`applicationName` : nom de l'application dans laquelle réside le projet.

Retour

Tableau d'objets Evaluation comportant un objet pour chaque projet correspondant au nom spécifié.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les évaluations.

AssessmentResults.getName

Signature

```
public String getName()
```

Description

Renvoie le nom de cet objet `AssessmentResults`.

Retour

Chaîne contenant le nom de cet objet `AssessmentResults`.

AssessmentResults.getMessages

Signature

```
public Message[] getMessages()
```

Description

Permet l'accès à tous les messages de statut s'étant affichés sur la console lors de l'exécution de l'évaluation dans AppScan Source for Analysis.

Retour

Tableau d'objets Message.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les messages.

Call

La classe `com.ouncelabs.sdk.assessment.Call` représente un noeud dans une trace AppScan Source.

Call.getCalls

Signature

```
public Call[] getCalls()
```

Description

Permet d'accéder aux appels effectués dans le contexte de cet appel.

Retour

Tableau d'objets Call (appel) d'après leur ordre d'appel

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les appels.

Call.getFilename

Signature

```
public String getFilename()
```

Description

Permet d'accéder au nom de fichier dans lequel cet appel est intervenu.

Retour

Nom de fichier de cet appel.

Call.getLineNumber

Signature

```
public int getLineNumber()
```

Description

Permet d'accéder au numéro de ligne sur laquelle cet appel est intervenu.

Retour

Numéro de ligne de cet appel.

Call.getColumnNumber

Signature

```
public int getColumnNumber()
```

Description

Cet appel permet d'accéder au numéro de la colonne sur laquelle cet appel est intervenu.

Retour

Numéro de colonne d'où provient l'appel.

Call.getSignature

Signature

```
public String getSignature()
```

Description

Permet d'accéder à la signature de l'API appelée.

Retour

Signature de cet appel.

Call.getSrcContext

Signature

```
public String getSrcContext()
```

Description

Permet d'accéder au contexte source de l'appel.

Retour

Contexte source de cet appel.

Call.getMethodName

Signature

```
public String getMethodName()
```

Description

Permet d'accéder au nom de méthode appelé.

Retour

Nom de méthode pour cet appel.

Call.getClassName

Signature

```
public String getClassName()
```

Description

Permet d'accéder au nom de la classe à laquelle appartient la méthode appelée.

Retour

Nom de classe de la méthode appelée.

Call.getTraceType

Signature

```
public TraceType getTraceType()
```

Description

Permet d'accéder au type de trace pour cet appel. Le type de trace, par exemple, peut spécifier que l'appel est soit une source, soit un collecteur.

Retour

Objet `TraceType` qui représente le type de trace pour l'appel.

ClassificationType

La classe `com.ouncelabs.sdk.assessment.ClassificationType` représente la classification d'une constatation.

Membres statiques

- `ClassificationType.Vulnerability`
- `ClassificationType.Type1`
- `ClassificationType.Type2`
- `ClassificationType.Unknown`

ClassificationType.value

Signature

```
public int value()
```

Description

Permet d'accéder à la valeur de cet objet `ClassificationType`. La valeur renvoyée correspond à l'une des valeurs des membres statiques.

Retour

Valeur de classification pour cet objet `ClassificationType`.

DateProximityUnit

Lorsque `com.ouncelabs.sdk.assessment.DateProximityUnit` est couplé avec `dateProximityDuration`, indique l'unité (jours, semaines, etc.) d'après laquelle effectuer le comptage. Requis lorsque `dateProximityDuration` est spécifié. Les unités valides sont décrites dans cette section.

Membres statiques

- `DateProximityUnit.Hour`
- `DateProximityUnit.Day`
- `DateProximityUnit.Week`
- `DateProximityUnit.Month`
- `DateProximityUnit.Year`

DateProximityUnit.value

Signature

```
public EnumType value()
```

Description

Permet d'accéder à la valeur de cet objet `DatProximityUnit`. La valeur renvoyée correspond à l'une des valeurs des membres statiques.

Retour

Valeur de date pour cet objet `DateProximityUnit`.

Factory

`com.ouncelabs.sdk.Factory` fournit des méthodes pour l'initialisation, la connexion et l'ouverture d'évaluations. Cette classe est le point d'entrée dans l'API d'accès aux données.

Factory.init

Signature

```
public void init(String installDir, boolean acceptInvalidSSLCerts)
```

Description

Initialise l'API d'accès aux données. Vous devez appeler `Factory.init` avant d'appeler toute autre méthode.

Paramètres

- `installDir` : chemin du répertoire d'installation de AppScan Source (voir «Répertoire d'installation par défaut», à la page 151).
- `acceptInvalidSSLCerts` : définit le booléen sur true lorsque vous souhaitez accepter les certificats SSL non valides (les certificats SSL ne requièrent alors aucune validation).

Exception renvoyée

`OunceException`, en cas d'échec de l'initialisation. Ceci ne survient généralement que si vous avez spécifié un répertoire erroné.

Factory.shutdown

Signature

```
public void shutdown()
```

Description

Quitte l'API d'accès aux données. Si `Factory.init` a été appelée, vous devez appeler `Factory.shutdown` pour que l'API d'accès aux données soit fermée correctement. Si vous l'omettez, `OunceScanner.exe` peut continuer à s'exécuter en tant que processus vagabond.

Factory.clearCache

Signature

```
public void clearCache()
```

Description

Cet appel efface le cache de l'API d'accès aux données. Il doit être utilisé périodiquement lors du traitement d'un grand nombre de résultats de constatations quand la mémoire est utilisée de façon excessive.

Factory.login

Signature

```
public void login(String nom_d'utilisateur,
String mot_de_passe, String nom_d'hôte[int :port])
```

Description

Ouvre une session sur le serveur AppScan Enterprise Server s'exécutant à l'URL spécifiée ou sur l'hôte dont le nom est indiqué. Vous devez vous connecter pour pouvoir publier les évaluations dans la base de données AppScan Source.

Paramètres

- `nom_d'utilisateur` : nom de l'utilisateur se connectant.
- `mot_de_passe` : mot de passe de l'utilisateur se connectant.
- `nom_d'hôte` : adresse URL ou nom d'hôte du serveur AppScan Enterprise Server auquel vous désirez vous connecter. Le `nom_d'hôte` peut être une adresse IP ou un nom de domaine.

En cas de besoin, vous pouvez spécifier le numéro de port AppScan Enterprise Server en ajoutant la mention `:<port>`. Par exemple, `public void login("hwall", "shhh2008", "MyHost:2880")`.

Exception renvoyée

`OunceException`, en cas d'échec de la connexion.

Factory.openAssessment

Signature

```
public AssessmentResults openAssessment(String nom_chemin)
```

Description

Ouvre l'évaluation spécifiée d'après son nom de chemin.

Paramètres

`nom_chemin` : chemin d'accès à un fichier d'évaluation.

Retour

Objet `AssessmentResults` représentant l'évaluation spécifiée par `nom_chemin`.

Exception renvoyée

- `FileNotFoundException`, si le nom de fichier spécifié n'existe pas ou ne peut pas être ouvert.
- `OunceException`, si l'API d'accès aux données ne peut pas charger le fichier d'évaluation.

Factory.getPublishedAssessments

Signature

```
public AssessmentResults[] getPublishedAssessments(AssessmentFilter filter)
```

Description

Permet d'accéder aux évaluations publiées. Elle reçoit un objet `AssessmentFilter` que vous pouvez utiliser pour spécifier l'ensemble d'évaluations publiées à extraire.

Paramètres

`filtre` : objet `AssessmentFilter` spécifiant l'ensemble d'évaluations publiées à extraire.

Retour

Tableau des objets `AssessmentResults` correspondant au filtre spécifié.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les résultats de l'évaluation.

`Factory.diffAssessments`

Signature

```
public AssessmentDiff diffAssessments
(AssessmentResults result1, AssessmentResults result2)
```

Description

Trouve la différence entre deux résultats d'évaluation.

Exception renvoyée

`OunceException`.

`Finding`

La classe `com.ouncelabs.sdk.assessment.Finding` représente une constatation individuelle au sein d'une évaluation. Elle permet d'accéder à toutes les données associées à une constatation, comme sa classification et sa gravité.

`Finding.getFilename`

Signature

```
public String getFilename()
```

Description

Permet d'accéder au nom de fichier dans lequel cette constatation a été détectée.

Retour

Chaîne contenant le chemin d'accès absolu du fichier dans lequel cette constatation a été détectée.

Finding.getLineNumber

Signature

```
public int getLineNumber()
```

Description

Permet d'accéder au numéro de ligne sur laquelle cette constatation a été détectée.

Retour

Numéro de ligne de cette constatation.

Finding.getApiName

Signature

```
public String getApiName()
```

Description

Permet d'accéder au nom de l'API pour cette constatation.

Retour

Nom de l'API de cette constatation.

Finding.getCallerName

Signature

```
public String getCallerName()
```

Description

Permet d'accéder à la signature de la méthode qui contenait l'appel de l'API que cette constatation représente.

Retour

Signature de l'appelant.

Finding.getClassification

Signature

```
public ClassificationType getClassification()
```

Description

Permet d'accéder à la classification actuelle de cette constatation.

Retour

Objet `ClassificationType` qui représente la classification de la constatation.

Finding.getSeverity

Signature

```
public SeverityType getSeverity()
```

Description

Obtient la valeur actuelle de la gravité dans la constatation.

Retour

Objet SeverityType représentant la gravité de la constatation. SeverityType est décrit dans la rubrique «SeverityType.value», à la page 92.

Finding.getVulnerabilityType

Signature

```
public String getVulnerabilityType()
```

Description

Permet d'accéder au type de vulnérabilité actuel de la constatation.

Retour

Type de vulnérabilité de la constatation. Les types de vulnérabilité (par exemple, Dépassement de mémoire tampon ou Injection SQL) sont décrits dans la Base de connaissances de sécurité AppScan Source Security.

Finding.getOriginalClassification

Signature

```
public ClassificationType getOriginalClassification()
```

Description

Permet d'accéder à la classification originale de la constatation. Si la constatation n'a pas été modifiée, getOriginal est identique à getClassification.

Retour

Objet ClassificationType représentant la classification originale d'une constatation si cette classification a été modifiée.

Finding.getOriginalSeverity

Signature

```
public SeverityType getOriginalSeverity()
```

Description

Obtient la valeur originale de la gravité de la constatation. Si la gravité n'a pas été modifiée, getOriginalSeverity est identique à getSeverity.

Retour

Objet `SeverityType` représentant la gravité originale d'une constatation lorsque cette gravité a été modifiée.

`Finding.getOriginalVulnerabilityType`

Signature

```
public String getOriginalVulnerabilityType()
```

Description

Permet d'accéder au type de vulnérabilité original de la constatation. Si la constatation n'a pas été modifiée, `getOriginalVulnerabilityType` est identique à `getVulnerability`.

Retour

Objet `VulnerablilityType` représentant le type de vulnérabilité original d'une constatation lorsque ce type a été modifié.

`Finding.getModifiedClassification`

Signature

```
public ClassificationType getModifiedClassification()
```

Description

Permet d'accéder à la classification modifiée de la constatation.

Retour

Objet `ClassificationType` qui représente la classification d'une constatation lorsque cette classification a été modifiée (Null en l'absence de correspondance).

`Finding.getModifiedSeverity`

Signature

```
public SeverityType getModifiedSeverity()
```

Description

Extrait la valeur modifiée de la gravité. Identique à `getSeverity()` si elle a été modifiée.

Retour

Objet `SeverityType` représentant la gravité d'une constatation lorsque cette gravité a été modifiée (Null en l'absence de correspondance).

`Finding.getModifiedVulnerabilityType`

Signature

```
public String getModifiedVulnerabilityType()
```

Description

Permet d'accéder au type de vulnérabilité modifié d'une constatation.

Retour

Objet `VulnerablilityType` représentant le type de vulnérabilité d'une constatation lorsque ce type a été modifié (Null en l'absence de correspondance).

`Finding.getSrcContext`

Signature

```
public String getSrcContext()
```

Description

Permet d'accéder au contexte source de cette constatation.

Retour

Contexte source de cette constatation.

`Finding.getDefectSubmissionUser`

Signature

```
public String getDefectSubmissionUser()
```

Description

Permet d'accéder au nom d'utilisateur ayant soumis en dernier cette constatation à un système de suivi des défauts externe. Notez qu'il s'agit du nom d'utilisateur du système de suivi des défauts, qui peut être différent du nom d'utilisateur AppScan Source.

Retour

Nom d'utilisateur lors de la dernière soumission de cette constatation au système de suivi des défauts.

`Finding.getDefectSubmissionDate`

Signature

```
public String getDefectSubmissionDate()
```

Description

Permet d'accéder à la chaîne de date correspondant à la dernière soumission de cette constatation à un système de suivi des défauts externe.

Retour

Chaîne de date correspondant à la dernière soumission de cette constatation au système de suivi des défauts.

Finding.getDefectInfo

Signature

```
public String getDefectInfo()
```

Description

Permet d'accéder à l'ID du système de suivi des défauts externe attribué lors de la dernière soumission de cette constatation à un de ces systèmes.

Retour

Chaîne avec les ID de défaut attribués à cette constatation.

Finding.getProperties

Signature

```
public String[] getProperties()
```

Description

Cet appel permet d'accéder aux propriétés associées à la constatation. Ces propriétés fournissent des informations sur le type de vulnérabilité de la constatation et sur son API.

Retour

Tableau de propriétés :

- `Vulnerability.value` : type de vulnérabilité signalé dans les vues Constatation, tel que :

  ```
  Vulnerability.BufferOverflow
  Vulnerability.Injection.SQL
  ```

- `Mechanism.value` : mécanisme de sécurité d'API, utilisé pour définir des catégories de rapport personnalisées, tel que :

  ```
  Mechanism.AccessControl
  Mechanism.Cryptography
  ```

- `Technology.value` : technologie d'API, utilisée pour définir des catégories de rapport personnalisées, telle que :

  ```
  Technology.Communications.HTTP
  Technology.Database
  ```

- `Attribute.value` : balise de méthode d'API, telle que :

  ```
  Attribute.Deprecated
  Attribute.Modifier.Protected
  ```

Pour plus d'informations sur les valeurs de propriété, voir la Base de connaissances de sécurité AppScan Source Security.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les propriétés.

Finding.isExcluded

Signature

```
public boolean isExcluded()
```

Description

Indique si cette constatation est exclue de l'évaluation. Les constatations exclues ne sont pas prises en compte dans les statistiques de l'évaluation.

Retour

Si la constatation est exclue, `true`, sinon, `false`.

`Finding.getTrace`

Signature

```
public Trace getTrace()
```

Description

Permet d'accéder à la trace AppScan Source éventuelle associée à cette constatation.

Retour

Objet Trace pour la constatation s'il existe, Null autrement.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire la trace.

`Finding.getNotes`

Signature

```
public String getNotes()
```

Description

Permet d'accéder aux notes sur une constatation.

Retour

Notes éventuelles sur la constatation, Null en leur absence.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les notes.

`Trace`

La classe `com.ouncelabs.sdk.assessment.Trace` représente la trace AppScan Source d'une constatation.

`Trace.getCalls`

Signature

```
public Call[] getCalls()
```

Description

Permet d'accéder à l'appel root de la trace AppScan Source.

Retour

Tableau d'objets `Call`. Actuellement, contient toujours un seul objet.

Exception renvoyée

`OunceException`, si l'API d'accès aux données ne peut pas extraire les appels.

SeverityType

La classe `com.ouncelabs.sdk.assessment.SeverityType` représente la gravité d'une constatation.

Membres statiques

Les types de gravité sont les suivants :

- `SeverityType.High`
- `SeverityType.Medium`
- `SeverityType.Low`
- `SeverityType.Info`
- `SeverityType.Unknown`

SeverityType.value

Signature

```
public int value()
```

Description

Permet d'accéder à la valeur de cet objet `SeverityType`. La valeur renvoyée correspond à l'une des valeurs des membres statiques.

Retour

Valeur de gravité de cet objet `SeverityType`.

OunceException

`com.ouncelabs.sdk.assessment.OunceException` est la classe d'exception utilisée par l'API d'accès aux données. Pour plus d'informations sur une exception, appelez `getMessage()` sur l'exception concernée.

Chapitre 5. Plug-in Ounce/Maven

Cette section décrit le plug-in Ounce/Maven, qui utilise l'outil de génération Maven d'Apache pour intégrer AppScan Source dans le flux de travaux Maven.

Le plug-in Ounce/Maven vous permet de créer des fichiers d'application et de projet AppScan Source. Si AppScan Source for Automation est également installé, utilisez Ounce/Maven pour exécuter des examens de sécurité du code source et générer des rapports de sécurité exhaustifs.

Pour plus d'informations sur AppScan Source for Automation, voir Chapitre 6, «AppScan Source for Automation», à la page 99.

Pour plus d'informations sur la famille de produits AppScan Source, voir http://www.ibm.com/software/rational/products/appscan/source/.

Installation d'Ounce/Maven

Avant de commencer

Les composants suivants sont prérequis pour l'installation et l'utilisation d'Ounce/Maven :

- Apache Maven version 2.x ou versions ultérieures : pour plus d'informations sur le déploiement et l'utilisation des plug-in Maven, reportez-vous au site Web Apache Maven Project à l'adresse http://maven.apache.org/.
- Pour effectuer un examen à l'aide du plug-in Maven, vous avez besoin d'AppScan Source for Automation : Pour plus d'informations, voir Chapitre 6, «AppScan Source for Automation», à la page 99.

Pourquoi et quand exécuter cette tâche

Après l'installation et la configuration de Maven, Ounce/Maven est téléchargé la première fois que vous le référencez.

La documentation du site Ounce/Maven comprend des descriptions des objectifs Ounce/Maven, de leurs paramètres, des exemples, des notes d'utilisation et des exemples détaillés.

La documentation du site est disponible à l'adresse http://mojo.codehaus.org/ounce-maven-plugin.

Procédure

1. Si ce n'est déjà fait, installez Maven depuis http://maven.apache.org/. Suivez les instructions du site Web Maven.
2. Effectuez l'une des opérations suivantes :
   - Modifiez un fichier Maven `pom` pour y inclure un ou plusieurs objectifs Ounce/Maven, comme décrit dans la documentation du site Ounce/Maven.
   - Appelez un ou plusieurs objectifs Ounce/Maven depuis la ligne de commande, comme décrit dans la documentation du site Ounce/Maven.

Utilisation d'Ounce/Maven

Le plug-in Ounce/Maven vous permet d'utiliser Ounce/Maven afin de créer des projets et des applications AppScan Source, d'examiner les applications, de publier les évaluations résultantes et de générer des rapports AppScan Source. Spécifiez les objectifs Ounce/Maven et leurs paramètres comme pour les autres plug-in Maven.

Vous pouvez appeler les commandes Ounce/Maven de deux façons :

- En utilisant un fichier `pom` (fichier de génération Maven) : le fichier `pom` vous permet de créer des fichiers d'application et de projet AppScan Source dans le cadre de votre génération. Après avoir installé Ounce/Maven, vous pouvez modifier un fichier Maven `pom` en spécifiant les objectifs `ounce:application` et `ounce:project-only` requis pour vos tâches AppScan Source.
- Depuis la ligne de commande : appelez les objectifs `ounce:project`, `ounce:scan` et `ounce:report` depuis la ligne de commande pour créer des fichiers de projet AppScan Source (ou pour remplacer des paramètres de fichier de projet dans le fichier `pom`), lancer des examens AppScan Source, publier des évaluations et générer des rapports AppScan Source.

Chaque objectif Ounce/Maven inclut un certain nombre de paramètres.

- Pour plus d'informations sur les objectifs Ounce/Maven, voir «Objectifs Ounce/Maven», à la page 95.
- Pour plus d'informations sur les paramètres de chaque objectif, reportez-vous à la documentation Ounce/Maven sur le site http://mojo.codehaus.org/ounce-maven-plugin.

Scénarios Ounce/Maven

Ounce/Maven vous permet de :

- Créer des fichiers de projet et d'application AppScan Source
- Examiner des applications et publier les résultats (évaluations) dans la base de données AppScan Source
- Générer des rapports à partir des résultats des évaluations
- Intégrer les rapports AppScan Source sur le `site` cible

Cette section décrit succinctement ces tâches. Pour plus d'informations sur les concepts propres à AppScan Source et sur les langages, voir le manuel *AppScan Source for Analysis - Guide d'utilisation*.

Création de fichiers d'application et de projet

AppScan Source utilise un modèle d'application et de projet pour gérer les projets, comme suit :

- **Application** : une application contient des données de configuration et des informations personnalisables, ainsi qu'une liste des projets dans cette application.
- **Projet** : un projet consiste d'un ensemble de fichiers (y compris du code source) et des informations associées (telles que des données de configuration). Pour analyser un projet, celui-ci doit faire partie d'une application.

L'objectif `ounce:application` crée des fichiers d'application AppScan Source (`.paf`), tandis que les objectifs `ounce:project` et `ounce:project-only` créent des fichiers de projet AppScan Source (`.ppf`).

Pour plus d'informations sur les autres formats des fichiers de projet et d'application, voir le manuel *AppScan Source for Analysis - Guide d'utilisation*.

Analyse d'applications

Un examen AppScan Source analyse le code source pour détecter les vulnérabilités de la sécurité. Le résultat d'un examen est une évaluation, laquelle est un fichier XML.

Remarque : Pour plus d'informations sur les fonctionnalités de AppScan Source, voir le manuel *IBM Security AppScan Source for Analysis - Guide d'utilisation*.

Utilisez l'objectif `ounce:scan` depuis la ligne de commande pour examiner l'application et ses projets et, si vous le souhaitez, générer un rapport à partir de l'évaluation.

A l'issue d'une analyse, Ounce/Maven vous permet de :

- Publier l'évaluation dans la base de données AppScan Source. Les résultats de l'évaluation sont ainsi accessibles autres utilisateurs ayant accès à la base de données et munis des privilèges requis.
- Générer un rapport.

Génération de rapport

Dans de nombreux cas, la sortie souhaitée de AppScan Source est un rapport détaillé fournissant des données sur les évaluations. L'objectif `ounce:report` vous permet de générer ces rapports à partir d'évaluations nouvelles ou antérieures. L'objectif `ounce:report` peut, en cas de besoin, examiner une application, puis publier son évaluation ou générer des rapports sur l'évaluation. A la différence d'`ounce:scan`, vous pouvez utiliser `ounce:report` pour générer des rapports depuis des évaluations existantes.

Intégration de rapports avec le site cible

Pour intégrer `ounce:report` avec le site cible, ajoutez le plug-in Ounce/Maven à la section de génération de rapports du fichier `pom`. La cible de site exécute `ounce:report` et le rapport devient partie intégrante de la documentation du site pour votre application.

Lors d'une intégration avec le site cible, vous n'avez pas besoin de spécifier de paramètres supplémentaires. Vous pouvez spécifier le type de rapport en sélectionnant l'une des valeurs décrites dans la rubrique «Valeurs de reportType», à la page 97. Si vous n'en spécifiez pas, le rapport par défaut est du type `Constatations`.

Objectifs Ounce/Maven

Ounce/Maven fournit les objectifs suivants pour exécuter des fonctions AppScan Source :

- `ounce:application` : génère un fichier d'application AppScan Source contenant des références à tous les projets enfant définis dans le fichier `pom`. Une application est requise pour l'examen (et donc la génération de rapport).

- `ounce:project` : crée un ou plusieurs fichiers de projet AppScan Source selon le nombre de projets enfant Maven. L'objectif `ounce:project` est destiné à s'exécuter depuis la ligne de commande et intègre la génération Maven dans l'objectif.
- `ounce:project-only` : crée un ou plusieurs fichiers de projet AppScan Source selon le nombre de projets enfant Maven. L'objectif `ounce:project-only` est fourni afin d'intégrer la création de fichiers de projet AppScan Source dans le cycle de vie de génération Maven.
- `ounce:scan` : examine une application. Vous pouvez éventuellement publier l'évaluation ou générer un rapport. Exécutez l'objectif `ounce:scan` depuis la ligne de commande.
- `ounce:report` : génère un rapport AppScan Source. Si vous avez besoin d'actualiser les résultats, lancez un examen avant de générer le rapport. Exécutez l'objectif `ounce:report` depuis la ligne de commande.

Remarque :

- Pour rendre portables vos projets d'application et projet, créez des variables de chemin mappant les chemins de fichiers à leurs emplacements.
- Pour consulter des exemples d'utilisation d'objectifs Ounce/Maven, reportez-vous à la documentation Ounce/Maven sur le site http://mojo.codehaus.org/ounce-maven-plugin/.

`ounce:application`

Description

Génère un fichier d'application AppScan Source contenant des références à tous les projets enfant définis dans le fichier pom. Une application est requise pour l'examen et la génération de rapport. Bien que `ounce:application` ne crée pas de fichiers de projet AppScan Source, vous pouvez exécuter `ounce:project-only` afin de créer le projet depuis le même fichier de génération.

`ounce:project`

Description

Utilisez `ounce:project` depuis la ligne de commande. L'objectif `ounce:project` crée un ou plusieurs fichiers de projet AppScan Source selon le nombre de projets enfant Maven.

Utilisation d'`ounce:project`

Lorsque vous appelez l'objectif `ounce:project` depuis la ligne de commande, il construit automatiquement les dépendances requises. Si aucun fichier d'application n'existe, `ounce:project` le construit.

Tous les paramètres `ounce:project` sont facultatifs. Le nom du projet est déterminé par l'élément Maven `artifactId` et vous ne pouvez pas le remplacer.

Exécutez Maven avec un objectif `ounce:project` depuis le répertoire contenant le fichier `pom.xml` de niveau supérieur (ou, si seulement un sous-ensemble doit être généré, depuis un sous-répertoire contenant un fichier `pom.xml` enfant).

ounce:project-only

Description

L'objectif ounce:project-only est utilisé depuis un fichier pom. L'objectif ounce:project-only crée un ou plusieurs fichiers de projet AppScan Source selon le nombre de projets enfant Maven.

Utilisation d'ounce:project-only

Tous les paramètres ounce:project-only sont facultatifs. Le nom du projet est déterminé par l'élément Maven artifactId et vous ne pouvez pas le remplacer.

ounce:scan

Description

L'objectif ounce:scan génère un examen d'une application donnée, suivie éventuellement d'un rapport sur l'évaluation qui en découle. Exécutez ounce:scan depuis la ligne de commande.

Remarque : L'objectif ounce:scan n'est pas destiné à faire partie du cycle de vie de génération Maven.

Utilisation d'ounce:scan

Lorsque vous spécifiez ounce:scan, vous pouvez demander à ce qu'Ounce/Maven génère un rapport immédiatement à la suite de l'analyse. Dans ce cas, spécifiez les paramètres du rapport comme décrit dans «Valeurs de reportType» et «Valeurs de reportOutputType», à la page 98. Si vous spécifiez l'élément reportType, vous devez également spécifier reportOutputType et reportOutputPath.

Remarque : L'objectif ounce:scan crée ou met à jour, le cas échéant, les fichiers d'application et de projet.

ounce:report

Description

L'objectif ounce:report génère un rapport à partir d'une évaluation. Si vous ne spécifiez pas une évaluation existante, ounce:report exécute ounce:scan avant de générer le rapport. Exécutez ounce:report depuis la ligne de commande.

Spécifiez les paramètres du rapport comme décrit dans «Valeurs de reportType» et «Valeurs de reportOutputType», à la page 98. Si vous spécifiez l'élément reportType, vous devez également spécifier reportOutputType et reportOutputPath.

Valeurs de reportType

- Rapport sur les constatations :
  - Constatations par groupement
  - Constatations par API
  - Constatations par classification
  - Constatations

  - Activité DTS
  - Constatations par type
  - Constatations par énumération des faiblesses courantes (CWE)
  - Constatations par fichier
- Rapport AppScan Source :
  - CWE SANS Top 25 2011
  - DISA Application Security and Development STIG V3R10
  - OWASP Mobile Top 10
  - OWASP Top 10 2013
  - PCI Data Security Standard V3.2
  - Profil de sécurité logicielle
- Rapport personnalisé, s'il en existe.

Valeurs de `reportOutputType`

- Spécifiez l'un des formats suivants pour ce rapport :
  - `html` : génère un rapport au format HTML et l'affiche en ligne.
  - `zip` : crée un fichier ZIP contenant tous les éléments du rapport HTML.
- Pour les rapports au format PDF, vous pouvez spécifier leur niveau de détail :
  - `pdf-summary` (pdf récapitulatif) : contient des comptages pour chaque groupe de rapport personnalisé
  - `pdf-detailed` (pdf-détaillé) : contient des comptages pour chaque API de chaque propriété d'une vulnérabilité
  - `pdf-comprehensive` (pdf-complet) : contient des tableaux couvrant chaque constatation pour chaque API
  - `pdf-annotated` (pdf-annoté) : contient toutes les constatations, les notes éventuelles ajoutées aux constatations et les fragments de code pertinents
  - `pdf-annotated` (pdf-annoté) : génère un rapport annoté au format de fichier PDF.

Chapitre 6. AppScan Source for Automation

Automation Server (`ounceautod`) vous permet d'automatiser des aspects clés du flux de travaux AppScan Source et d'intégrer la sécurité aux environnements de génération lors du cycle de vie de développement de logiciels. Automation Server vous permet de placer en file d'attente des requêtes d'examen et de publication d'évaluations, et de générer des rapports sur la sécurité du code d'application.

Les requêtes sont soumises au serveur via l'exécutable de ligne de commande AppScan Source for Automation (`ounceauto`). Lors du traitement de requêtes, Automation Server s'exécute en tant que client du serveur AppScan Enterprise Server associé et ne peut se connecter qu'à une seule instance de AppScan Enterprise Server. Il attend des connexions de l'hôte local uniquement en écoutant le port TCP (par défaut, le port 13205).

- Sur les systèmes Windows, Automation Server s'exécute en tant que service **IBM Security AppScan Source Automation**.
- Sur les systèmes Linux, il s'exécute en tant que démon :
  - Pour arrêter le démon, lancez la commande suivante : `/etc/init.d/ounceautod stop`
  - Pour démarrer le démon, lancez la commande suivante : `/etc/init.d/ounceautod start`

Automation Server traite les demandes au nom de l'utilisateur AppScan Source spécifié et hérite par conséquent des autorisations de cet utilisateur. Cet ID utilisateur doit disposer des autorisations dont il a besoin compte tenu des commandes qu'il doit exécuter. Par exemple, si l'ID utilisateur doit exécuter la commande `PublishAssessment`, vous pouvez l'autoriser à publier et à enregistrer des autorisations sans pour autant qu'il ait besoin d'une autorisation d'examen (reportez-vous à la section *Administration de AppScan Source* du manuel *AppScan Source - Guide d'installation et d'administration* pour plus de détails). La soumission d'une requête àAutomation Server ne requiert pas de données d'identification utilisateur.

Spécification des données d'identification pour la connexion à Automation Server depuis la ligne de commande

Si vous n'avez pas spécifié de données d'identification Automation Server lors du processus d'installation (comme décrit dans le manuel IBM Security AppScan Source - Guide d'installation et d'administration), vous devez configurer le Automation Server après l'installation afin qu'il s'exécute en tant qu'utilisateur AppScan Source.

Pourquoi et quand exécuter cette tâche

Dans cette rubrique de tâches,

- `<rép_install>` représente l'emplacement de votre installation AppScan Source.
- `user_id` correspond à l'utilisateur avec lequel Automation Server s'authentifie lors du traitement d'une requête. Cet utilisateur doit être défini dans AppScan Source avec les autorisations requises.

Remarque : Si l'utilisateur n'existe pas encore, vous devrez le spécifier dans le panneau d'installation ou via la ligne de commande, puis le créer manuellement (post-installation) à l'aide de AppScan Source for Analysis ou de l'interface de ligne de commande (CLI) d'AppScan Source. Pour créer le nouvel utilisateur depuis la ligne de commande, utilisez la commande «newuser (nu)», à la page 44. Lors de la création du nouvel utilisateur, prenez soin de spécifier le même nom d'utilisateur et le même mot de passe que ceux spécifiés pour la connexion au Automation Server. Les autres paramètres (tels que les autorisations) peuvent être configurés selon vos besoins.

- `password` correspond au mot de passe de l'utilisateur.
- L'option `--persist` préserve les données d'identification sur disque et crée un fichier de clés chiffré avec les nom d'utilisateur et mot de passe spécifiés.
- Windows uniquement : `common_name` est votre nom usuel (CN) CAC.
- Windows uniquement : `cn_certificate` est le nom usuel de l'émetteur de votre certificat.

Procédure

- **Sur les systèmes Windows** :
  - Si vous utilisez l'ID utilisateur et le mot de passe comme méthode d'authentification AppScan Enterprise Server, exécutez la commande suivante :

    ```
    <install_dir>\bin\ounceautod.exe -u <nom_util> -p <mot_de_passe> --persist
    ```
  - Si AppScan Enterprise Server est activé pour l'authentification CAC (Common Access Card), entrez la commande suivante :

    ```
    <install_dir>\bin\ounceautod.exe -u <common_name<cn_certificate>> --persist
    ```

    Remarque : Si votre nom usuel (`common_name`) ou votre certificat CN (`cn_certificate`) comporte des espaces, placez-les entre guillemets (`"common_name<cn_certificate>"`).
- **Sur les systèmes Linux** :
  1. Modifiez la variable `LD_LIBRARY_PATH` afin d'inclure le répertoire `<install_dir>/bin` (où `<rép_install>` représente l'emplacement de votre installation AppScan Source), par exemple :

     ```
     export LD_LIBRARY_PATH=/opt/ibm/appscansource/
     bin:$LD_LIBRARY_PATH
     ```
  2. Lancez la commande suivante :

     ```
     <install_dir>/ibm/appscansource/bin/ounceautod
                   -u <ID_utilisateur> -p <mot_de_passe> --persist
     ```

Résultats

Une fois les données d'identification spécifiées, elles sont sauvegardées sur disque.

Fichier de configuration d'Automation Server

Le fichier de configuration d'`ounceautod`, `ounceautod.ozsettings`, spécifie les propriétés du démon du Automation Server. Le fichier `ounceautod.ozsettings` réside sous le répertoire `<data_dir>\config` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151).

Le fichier `ounceautod.ozsettings` contient les propriétés suivantes :

Propriété	Valeur	Description
ounceautod_max_ concurrent_requests	1 (valeur par défaut)	Nombre de requêtes simultanées autorisées.
ounceautod_accept_ssl	true (valeur par défaut)	Acceptation automatique des certificats SSL. Pour plus d'informations, voir «Certificats SSL pour AppScan Enterprise Server», à la page 40.
ounceautod_port	13205 (valeur par défaut)	Numéro de port sur lequel exécuter ounceautod. **Remarque :** Ce numéro de port doit correspondre à celui spécifié dans le fichier ounceauto.ozsettings sous le répertoire <data_dir>\config.
ounceautod_server_hostname	<nom d'hôte>:port	Nom d'hôte de l'ordinateur sur lequel AppScan Enterprise Server a été installé et port sur lequel il s'exécute sur cet ordinateur.

Si ce fichier est modifié, vous devez redémarrer le service du Automation Server, (décrit dans Chapitre 6, «AppScan Source for Automation», à la page 99).

Consignation sur Automation Server

Automation Server génère un fichier journal, ounceautod.log, qui consigne chaque requête émise vers ce serveur, ainsi que la réponse. Les entrées du journal sont consignées dans le fichier <data_dir>\logs\ounceautod.log (où <rép_données> est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151).

Automation Server consigne les événements suivants pour toutes les requêtes :

- Requête reçue
- Requête lancée
- Requête terminée
- Echec de requête

Chaque entrée du journal contient :

- Un horodatage de l'entrée au journal
- Le type d'événement
- Le type de requête
- L'ID de la requête
- L'appelant, s'il a été spécifié
- Des informations spécifiques sur la requête

Utilisation d'Ounceauto depuis la ligne de commande

L'interface de AppScan Source for Automation, ounceauto, émet des commandes vers le serveur spécifié dans les arguments de la ligne de commande, imprime la sortie éventuelle, et rend la main à l'utilisateur. A chaque requête est affecté un ID de requête et toutes les entrées au journal concernant la requête mentionnent cet ID à des fins d'identification.

Exécutez ounceauto depuis la ligne de commande comme suit :

```
ounceauto <nom commande> <arguments de la commande>
```

Les commandes Ounceauto comprennent :

- GenerateReport
- PublishAssessment
- PublishAssessmentASE
- ScanApplication
- Wait

GenerateReport

Description

Crée un rapport à partir d'une évaluation.

Syntaxe

```
ounceauto GenerateReport
-assessment <chemin de l'évaluation>
-type <type du rapport>
-output <format de sortie>
-file <emplacement de la sortie>
[-caller <appelant>]
[-includeSrcBefore <n>]
[-includeSrcAfter <n>]
[-includeTraceDefinitive]
[-includeTraceSuspect]
[-includeTraceCoverage]
```

- -assessment <chemin de l'évaluation> : chemin du fichier d'évaluation pour lequel générer le rapport.
- -type "<type de rapport>" : nom du type de rapport entre guillemets. Il peut s'agir de rapports sur les constatations, de rapports AppScan Source et de rapports personnalisés.

  Les types de rapport AppScan Source incluent :

  – Rapport sur les constatations :
    - Constatations par groupement
    - Constatations par API
    - Constatations par classification
    - Constatations
    - Activité DTS
    - Constatations par type
    - Constatations par énumération des faiblesses courantes (CWE)
    - Constatations par fichier

  – Rapport AppScan Source :

- CWE SANS Top 25 2011
- DISA Application Security and Development STIG V3R10
- OWASP Mobile Top 10
- OWASP Top 10 2013
- PCI Data Security Standard V3.2
- Profil de sécurité logicielle

– Rapport personnalisé, s'il en existe.

Lorsque vous entrez le type de rapport entre guillemets, entrez-le exactement tel qu'il est spécifié dans la liste ci-dessus, par exemple `Constatations par classification` ou `Profil de sécurité logicielle`.

- `-output <format de sortie>` : spécifiez l'un des formats de rapport suivants,
  – `html` : génère un rapport au format HTML et l'affiche en ligne.
  – `zip` : crée un fichier ZIP contenant tous les éléments du rapport HTML
  – Pour les rapports au format PDF, vous pouvez spécifier leur niveau de détail :
    - `pdf-summary` (pdf récapitulatif) : contient des comptages pour chaque groupe de rapport personnalisé
    - `pdf-detailed` (pdf-détaillé) : contient des comptages pour chaque API de chaque propriété d'une vulnérabilité
    - `pdf-comprehensive` (pdf-complet) : contient des tableaux couvrant chaque constatation pour chaque API
    - `pdf-annotated` (pdf-annoté) : contient toutes les constatations, les notes éventuelles ajoutées aux constatations et les fragments de code pertinents
    - `emplacement de sortie` : chemin de fichier pour l'écriture du rapport.
- `-file <emplacement de la sortie>` : indiquez le chemin et le nom du fichier dans lequel sauvegarder le rapport.
- `-caller <appelant>` : facultatif. Affectez un appelant à l'opération de génération du rapport. Il peut s'agir du nom d'un utilisateur réel, mais ceci n'est pas requis. Le nom de l'appelant est consigné au fichier journal `ounceauto`.
- `-includeSrcBefore <n>` : facultatif. Nombre de lignes du code source à inclure avant chaque constatation.
- `-includeSrcAfter <n>` : facultatif. Nombre de lignes du code source à inclure après chaque constatation.
- `-includeTraceDefinitive`: facultatif. Incluez les informations de trace dans le rapport pour les constatations définitives (voir «Classifications», à la page 153 pour plus de détails sur les classifications des constatations).
- `-includeTraceSuspect` : facultatif. Incluez les informations de trace dans le rapport pour les constatations suspectes.
- `-includeTraceCoverage` : facultatif. Incluez les informations de trace dans le rapport pour les constatations de couverture.

Valeur renvoyée

ID de la requête si celle-ci a abouti ou -1 si sa soumission a échoué.

Exemples

- Générez un rapport Constatations par API dans un fichier au format HTML. Dans le rapport, incluez les informations de trace pour les constatations définitives :

```
ounceauto generatereport -assessment C:\Ounce\Data\Webgoat.ozasmt
-type "Constatations par API" -output html
-file C:\reports\Webgoat_Findings.html
-includeTraceDefinitive
```

- Pour générer un rapport OWASP Top 10 2013 AppScan Source au format PDF, entrez la commande suivante :

```
ounceauto generatereport -assessment C:\Ounce\Data\Webgoat.ozasmt
-type "OWASP Top 10 2013" -output pdf-annotated
-file C:\Reports\Webgoat_OWASP.pdf
```

PublishAssessment

Description

Publie l'évaluation sélectionnée dans la base de données AppScan Source.

Syntaxe

ounceauto PublishAssessment - file <évaluation.ozasmt>
[-caller <appelant>]

- -file <fichier d'évaluation> : chemin d'accès complet d'une fichier d'évaluation.
- -caller <appelant> : facultatif. Affectez un appelant à l'opération de génération du rapport. Il peut s'agir du nom d'un utilisateur réel, mais ceci n'est pas requis. Le nom de l'appelant est consigné au fichier journal ounceauto.

Valeur renvoyée

ID de la requête si celle-ci a abouti ou -1 si sa soumission a échoué.

Exemple

Pour publier l'évaluation WebGoat_Internal, entrez la commande suivante :

```
ounceauto publishassessment -file C:\Ounce\Data\WebGoat_Internal.ozasmt
```

PublishAssessmentASE

Description

Publiez l'évaluation sélectionnée dans AppScan Enterprise Console.

Syntaxe

ounceauto PublishAssessmentASE -file <fichier_évaluation>
[-aseapplication <application_ase>] [-caller <appelant>]
[-folder <emplacement>] [-name <nom_évaluation_publiée>]
[-preventOverwrite]

- -file <fichier_évaluation> : requis. Chemin et nom du fichier d'évaluation.
- -aseapplication <application_ase> : Cette option est requise lors d'une connexion à AppScan Enterprise Server versions 9.0.3 et supérieures (sauf si vous désactivez l'exigence, comme indiqué ici). L'association à une application est facultative en cas de connexion aux versions antérieures de AppScan Enterprise Server. Utilisez cette option pour spécifier l'application Enterprise Console à associer à l'évaluation.
- -caller <appelant> : facultatif. Affectez un appelant à l'opération de génération du rapport. Il peut s'agir du nom d'un utilisateur réel, mais ceci n'est pas requis. Le nom de l'appelant est consigné au fichier journal ounceauto.

- `-folder <emplacement>` : facultatif. Cette option s'applique uniquement en cas de connexion à des versions AppScan Enterprise Server antérieures à la version 9.0.3. Indiquez le dossier Enterprise Console dans lequel publier l'évaluation. Si cet argument n'est pas utilisé, l'évaluation sera publiée sur votre dossier par défaut Enterprise Console.
- `-name <nom_évaluation_publiée>` : facultatif. Nom sous lequel l'évaluation est sauvegardée dans Enterprise Console. Si cet argument n'est pas utilisé, un nom est généré en fonction de l'application AppScan Source qui a été examinée pour générer l'évaluation (ce nom est préfixé par `AppScan Source :`).
- `-preventOverwrite` : facultatif. Incluez cet argument pour annuler la publication si une évaluation portant le même nom existe déjà sur le serveur.

Valeur renvoyée

ID de la requête si celle-ci a abouti ou -1 si sa soumission a échoué.

Exemple

Pour publier l'évaluation `WebGoat_Internal` dans AppScan Enterprise Server version 9.0.3, ou ultérieure, entrez la commande suivante :

```
ounceauto publishassessmentase -file C:\Ounce\Data\WebGoat_Internal.ozasmt
-aseapplication myapplication
```

Important :

Lors de la mise à niveau vers AppScan Source version 9.0.3.4, vous remarquerez les modifications suivantes :

- Lors de la publication d'une évaluation sur AppScan Enterprise Console, vous devez désormais associer celle-ci à une application dans AppScan Enterprise (si vous utilisez AppScan Enterprise Server version 9.0.3 ou supérieure). En conséquence, les scripts d'automatisation peuvent échouer s'ils ne contiennent pas l'association d'applications. AppScan Enterprise Server, l'association d'applications est requise pour bénéficier de ses fonctions de gestion des risques de sécurité pour les applications. Voir http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/c_overview.html.
- En outre, vous devez retirer le port de l'URL d'AppScan Enterprise.
  1. Dans AppScan Source for Analysis, cliquez sur **Editer** > **Préférences**.
  2. Dans les paramètres d'AppScan Enterprise Console, retirez le port de la zone **URL de la console Enterprise**.
- Après sa publication, l'évaluation n'est disponible que dans la vue Surveillance de AppScan Enterprise (dans les versions précédentes, elle était disponible dans les vues Examens). La migration vers cette vue est décrite à la rubrique http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/t_workflow_for_applications.html.

Il s'agit du résultat du changement de protocole de communication entre AppScan Source et AppScan Enterprise Server, requis pour la publication sur AppScan Enterprise Server lors de l'utilisation de l'authentification CAC (Common Access Card).

Si vous ne souhaitez pas publier des évaluations sur AppScan Enterprise Server lorsque l'authentification CAC est activée, ni tirer parti des fonctions de gestion des risques de sécurité pour les applications offertes par Enterprise Server, vous pouvez rétablir l'ancien protocole de communication de la manière suivante :

1. Ouvrez <datrép_données;\config\ounce.ozsettings (où <rép_données> est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151).
2. Dans ce fichier, repérez le paramètre suivant :

```
<Setting
  name="force_ase902_assessment_publish"
  value="false"
  default_value="false"
  description="Use ASE 9.0.2-style assessment publish"
  display_name="Use ASE 9.0.2-style assessment publish"
  type="boolean"
  read_only="true"
  hidden="true"
/>
```

3. Remplacez value="false" par value="true" puis sauvegardez le fichier.
4. Redémarrez le produit AppScan Source à partir duquel vous publiez les évaluations.

Lorsque ce paramètre est défini sur value="true" :

- Si vous associez une évaluation à une application dans AppScan Enterprise lors de la publication, l'évaluation sera disponible dans les vues Surveillance et Examens.
- Si vous n'associez pas l'évaluation à une application lors de la publication, celle-ci sera disponible dans la vue Examens.
- Vous ne pourrez pas publier des évaluations dans AppScan Enterprise Server lorsque l'authentification CAC est activée.

Pour plus d'informations, voir http://www.ibm.com/support/docview.wss?uid=swg21993010.

ScanApplication

Description

Examiner l'application spécifiée et réaliser d'autres actions liées à l'examen.

Syntaxe

```
ounceauto ScanApplication
-application <application_name>|
  -application_file <chemin_fichier_application>
[-name <nom_évaluation>]
[-scanconfig <nom_configuration_examen>]
[-save <nom_fichier>]
[-caller <appelant>]
[-publish]
[-clearcache]
[-report <type de rapport> <format de sortie>
<emplacement de la sortie>]
[-includeSrcBefore <n>]
[-includeSrcAfter <n>]
[-includeTraceDefinitive]
 on[-includeTraceSuspect]
[-includeTraceCoverage]
[-appserver_type]
[-include_all_lib_jars]
[-include_lib_jars]
[-no_ear_project]
```

- `-application <application_name>` **ou** `-application_file <chemin_fichier_application file>` : l'un de ces deux éléments est obligatoire .
  - Si vous spécifiez `-application <application_name>`, indiquez le nom de l'application à examiner.
  - Si vous spécifiez `-application_file <chemin_fichier_application>`, indiquez le chemin et le nom de fichier complet pour l'un de ces types de fichiers :
    - Fichiers d'application AppScan Source (`.paf`).
    - Espaces de travail Eclipse ou Rational Application Developer for WebSphere Software (RAD) (`.ewf`)

      Remarque : Les fichiers `.ewf` sont générés lorsque vous utilisez `openapplication` pour ouvrir un répertoire d'espace de travail (en spécifiant son chemin d'accès).
    - Fichiers `WAR` (`.war`)
    - Fichiers `EAR` (`.ear`)
    - Windows uniquement : fichiers Microsoft Visual C++ Workspace (`.dsw`)
    - Windows uniquement : Fichiers de solution Microsoft Visual Studio.NET (`.sln`)

    Remarque : Pour savoir quelles versions des fichiers importés sont prises en charge par AppScan Source for Analysis, AppScan Source for Automation et l'interface de ligne de commande d'AppScan Source, voir http://www.ibm.com/support/docview.wss?uid=swg27027486. Dans cette page, sélectionnez l'onglet de la version d'AppScan Source que vous utilisez, puis sélectionnez le composant AppScan Source que vous utilisez. Si AppScan Source prend en charge l'ouverture et l'examen de fichiers provenant d'autres environnements de développement, cette prise en charge est indiquée à la section **Compilateurs et langues** de l'onglet **Logiciels pris en charge**.
- `-name <nom évaluation>` : facultatif. Nom pour l'évaluation.
- `-scanconfig <nom_configuration_analyse>`: facultatif. Indiquez le nom de la configuration d'examen à utiliser pour l'examen. Si aucune configuration d'examen n'est spécifiée, c'est la configuration d'examen par défaut qui est utilisée.
- `-save <nom fichier>` : facultatif. Sauvegarder les résultats de l'évaluation dans ce fichier.
- `-caller <appelant>` : facultatif. Affecter un appelant à l'opération. Il peut s'agir du nom d'un utilisateur réel, mais ceci n'est pas requis. Le nom de l'appelant est consigné au fichier journal `ounceauto`.
- `-publish` : facultatif. Publier l'évaluation après l'examen.
- `-clearcache` : facultatif. Supprimer le cache d'analyse de vulnérabilité et les données de signature des règles personnalisées avant l'examen. Si vous avez activé l'analyse incrémentielle Java, l'examen sera un examen complet.
- `-report` : facultatif. Générer un rapport après l'examen.
  - Options obligatoires de la commande `-report` :
    - `<report type>` : Type de rapport. Il peut s'agir de rapports sur les constatations, de rapports AppScan Source et de rapports personnalisés. Reportez-vous aux options indiquées dans la rubrique «`GenerateReport`», à la page 102.
    - `<format de sortie>` : spécifie le format du rapport. Reportez-vous aux options indiquées dans la rubrique «`GenerateReport`», à la page 102.
    - `<output location>` : emplacement dans lequel sauvegarder le rapport.

  - Options facultatives de la commande `-report` :
    - `-includeSrcBefore <n>` : Nombre de lignes du code source à inclure avant chaque constatation.
    - `-includeSrcAfter <n>` : Nombre de lignes du code source à inclure avant chaque constatation.
    - `-includeTraceDefinitive` : inclure les informations de trace dans le rapport pour les constatations définitives (voir «Classifications», à la page 153 pour plus de détails sur les classifications des constatations).
    - `-includeTraceSuspect` : inclure les informations de trace dans le rapport pour les constatations suspectes.
    - `-includeTraceCoverage` : inclure les informations de trace dans le rapport pour les constatations de couverture d'examen.
- `-appserver_type` : Facultatif. Si l'application en cours d'ouverture inclut JavaServer Pages (par exemple, un fichier `WAR` ou `EAR`), utilisez ce paramètre pour indiquer le serveur d'application à utiliser pour la compilation de JSP. Spécifiez l'une des options suivantes, avec des guillemets :
  - `Tomcat 7`
  - `Tomcat 8`
  - `WebSphere 7.0`
  - `WebSphere 8.0`
  - `WebSphere 8.5`
  - `WebLogic 11g`
  - `WebLogic 12c`

  Remarque :
  - Avant de spécifier un serveur d'application, vérifiez qu'il a été configuré correctement dans les préférences AppScan Source for Analysis.
  - Si `-appserver_type` n'est pas utilisé, le compilateur JSP par défaut défini actuellement dans AppScan Source for Analysis sera utilisé pour la compilation JSP. Prêt à l'emploi, Tomcat 7 est le compilateur JSP par défaut.
- Pour les fichiers `WAR` :
  - `-include_all_lib_jars` : Utilisez ce paramètre pour prendre en compte toutes les bibliothèques du fichier `WAR` lors de l'examen.
  - `-include_lib_jars` : Utilisez ce paramètre pour préciser les bibliothèques du fichier `WAR` à prendre en compte lors de l'examen. Dans ce cas, entrez les bibliothèques sans leur chemin et séparez-les par une virgule si vous en entrez plusieurs.
- `-no_ear_project` : Lors de l'importation d'un fichier `EAR`, un projet destiné au stockage des bibliothèques partagées est créé automatiquement. En l'absence de bibliothèques partagées, le fichier est quand-même créé, mais reste vide. Ce paramètre permet de ne pas créer de projet pour le fichier `EAR`.

Valeur renvoyée

ID de la requête si celle-ci a abouti ou -1 si sa soumission a échoué.

Exemples

- Examinez l'application WebGoat, publiez-la annotez le journal avec comme appelant John Smith :

```
ounceauto scanapplication -application_file C:\WebGoat\WebGoat.paf
-publish -caller JohnSmith
```

- Examinez l'application WebGoat et créez un rapport Constatations sous le répertoire `C:\WebGoat`. Dans le rapport, incluez les informations de trace pour les constatations définitives :

  ```
  ounceauto scanapplication -application WebGoat
  -report Findings html C:\WebGoat\MyReport.html
  -includeTraceDefinitive
  ```

- Examinez un fichier WAR en ne prenant en compte que certaines bibliothèques :

  ```
  ounceauto scanapplication -application_file c:\mywar.war
  -include_lib_jars lib1.jar,lib2.jar
  ```

Wait

Description

Bloque les opérations jusqu'à ce que la requête spécifiée se termine.

Syntaxe

ounceauto wait -requestid <ID_requête>

`-requestid <ID_requête>` : ID de la requête dont attendre l'exécution.

Valeur renvoyée

Si la requête correspondant à `<ID_requête>` aboutit, la valeur renvoyée est 0, autrement -1.

Exemple

L'exemple Windows suivant illustre une commande d'examen d'un fichier d'application et d'attente d'achèvement de cet examen :

```
ounceauto scanapplication -application_file WG0.paf
ounceauto wait -requestid %errorlevel%
```

Sous Linux, l'équivalent de cet exemple est :

```
ounceauto scanapplication -application_file WG0.paf
ounceauto wait -requestid $?
```

Chapitre 7. API de gestion Framework for Frameworks

AppScan Source fournit un ensemble d'API Java qui permettent d'ajouter le support des frameworks (infrastructures) utilisés dans vos applications. Les classes et méthodes disponibles dans ces API vous permettent de prendre en compte des frameworks dont le support n'est pas fourni en standard.

Remarque : AppScan Source comporte un support intégré pour ces frameworks :

- Apache Struts 1 et 2
- Spring MVC 2.5 et 3
- ASP .NET MVC (Windows uniquement)
- Enterprise JavaBeans (EJB) 2
- ASP .NET (Windows uniquement)
- J2EE
- JavaServer Faces (JSF) 2
- .NET 4.5 (Windows uniquement)
- Jax - RS (V1.0 et V1.1)
- Jax - WS (V2.2)

L'introduction des frameworks modernes a eu pour effet de déplacer une grande part des informations ayant une incidence sur le comportement des applications à l'exécution. Siégeant auparavant dans le code source, ces informations sont à présent intégrées dans des fichiers de configuration et des annotations. Jusqu'à une époque récente, cela se traduisait par des angles morts durant les analyses statiques. Les équipes produit pouvaient créer des règles spécialement étudiées pour leurs applications, mais il n'existait pas de méta-infrastructure capable de décrire avec souplesse et de manière automatisée les activités de ces frameworks.

En utilisant les API Framework for Frameworks, vous pouvez facilement ajouter le support de nouveaux frameworks directement dans AppScan Source. Pour cela, les informations de configuration associées aux frameworks concernés sont traitées et renvoyées à AppScan Source via les API correspondantes.

Les API Framework for Frameworks sont incluses avec l'installation des produits suivants :

- AppScan Source for Automation
- AppScan Source for Analysis
- AppScan Source for Development

Les API sont installées dans `<install_dir>\walalib` (où `<rép_install>` représente l'emplacement de votre installation AppScan Source).

Un exemple d'archive de projet est installé dans `<data_dir>\samples\F4FEjbExample.zip` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151).

Remarque : Les noeuds de trace dont le nom de classe commence par `Appscan.Synthetic`, `Appscan.Synthetic.Validator` et `AppScan.Synthetic.Replacement` correspondent à des méthodes synthétisées par AppScan Source.

- Les méthodes `AppScan.Synthetic` permettent d'assembler des traces dans le code d'application qui utilisé des structures.
- Une méthode `AppScan.Synthetic.Validator` modélise la validation sous-jacente effectuée par l'exécution de la structure. Vous pouvez sélectionner une méthode valideur et la marquer comme **Validator** si nécessaire.
- Une méthode `AppScan.Synthetic.Replacement` indique qu'une méthode du code d'application a été remplacée par AppScan Source pour capturer le flux de données entre des composants disjoints (tels que les contrôleurs et les vues) de la structure.

Composants principaux de l'API Framework for Frameworks

F4FHandler

`F4FHandler` est la classe abstraite que tous les nouveaux gestionnaires de framework doivent étendre. Elle dessine l'enchaînement d'activités d'un gestionnaire de framework à travers des méthodes abstraites qui doivent être redéfinies. Elle fournit également l'accès aux deux autres composants de l'API, `F4FApp` et `F4FAction`.

F4FApp

Fournit des informations sur l'application analysée, notamment celles qui proviennent de sa configuration (telles que les racines des sources et la racine Web). Ce composant fournit également l'accès à chaque classe de l'application dans le but de déterminer les informations nécessaires telles que les annotations et la superclasse.

F4FAction

Fournit à AppScan Source des informations sur l'application analysée. En appelant les méthodes fournies dans cette API, il est possible de décrire au moteur d'examen différents aspects de l'application tels que ses points d'entrée et les connexions entre les méthodes abstraites et leurs implémentations.

Utilisation des API Framework for Frameworks

Cette section décrit les étapes requises lorsque vous utilisez les API Framework for Frameworks. Lorsque cela est nécessaire, les étapes contiennent des références à l'exemple fourni pour vous permettre de suivre celui-ci.

Dans cet exemple, nous allons effectuer les opérations suivantes :

- Création d'une classe qui étend `F4FHandler`
- Implémentation de deux méthodes abstraites de `F4FHandler`
- Création d'une archive Java (`.jar`) avec un fichier manifeste qui identifie votre classe de gestionnaire
- Exportation du fichier `.jar` vers le répertoire `waflgens` de AppScan Source
- «A propos de l'exemple», à la page 113

- «Importation du projet exemple dans Eclipse ou dans Rational Application Developer for WebSphere Software (RAD)»
- «Création d'une classe qui implémente `F4FHandler`», à la page 116
- «Création d'un fichier manifeste pour votre gestionnaire», à la page 117
- «Création d'un fichier JAR pour votre gestionnaire», à la page 117
- «Exportation du fichier JAR vers `waflgens`», à la page 120

A propos de l'exemple

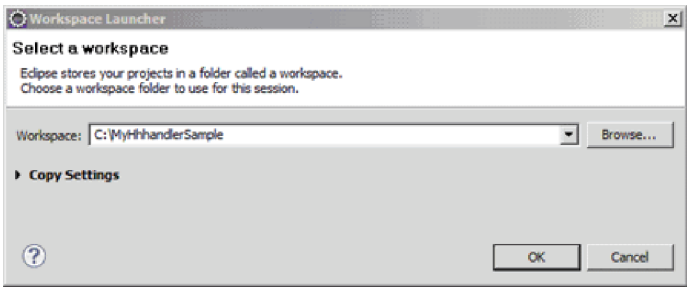
L'installation AppScan Source inclut un exemple de gestionnaire qui lit et fournit des informations Framework for Frameworks à propos des applications Enterprise Java Bean (EJB) 2 (également appelées Enterprise Java Beans ou plus simplement Beans).

EJB est un framework (infrastructure) qui s'efforce de faciliter la manipulation et la réutilisation de composants de logique métier. Chaque bean représente un composant métier et doit être associé à des classes d'interface permettant aux autres beans d'interagir avec lui. L'association entre ces classes d'interface et le bean se fait par le biais du fichier de configuration EJB de l'application (`EJB-jar.xml`). Il en résulte un "angle mort" pour le moteur d'analyse statique, car celui-ci ne perçoit pas ce type d'association dans le code source. EJB est donc un bon candidat à la gestion via Framework for Frameworks.

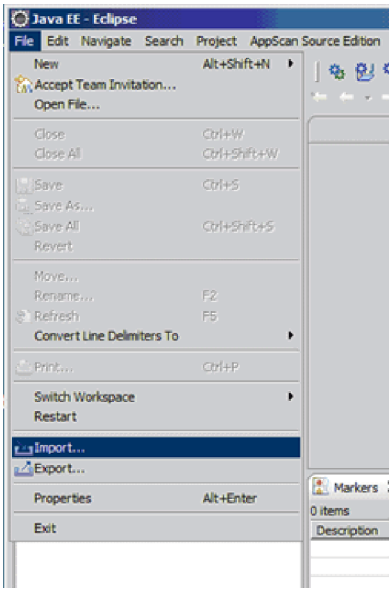
Importation du projet exemple dans Eclipse ou dans Rational Application Developer for WebSphere Software (RAD)

Procédure

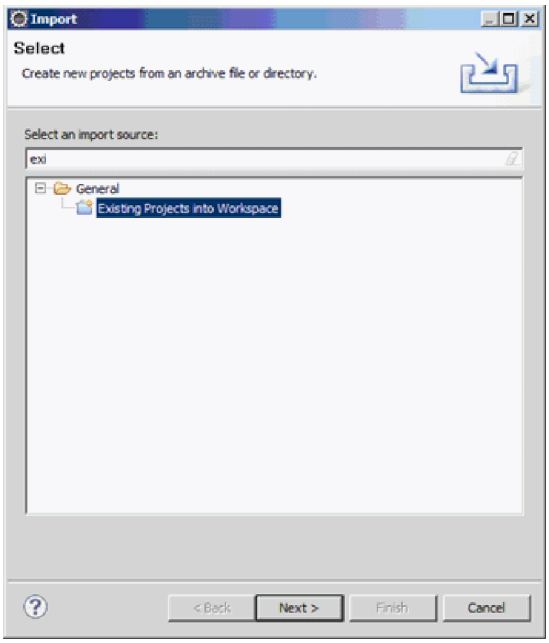
1. Démarrez Eclipse.
2. Créez un nouvel espace de travail pour votre projet de gestionnaire.



3. Sélectionnez **Fichier** > **Importer** sur la barre de menus principale du plan de travail.



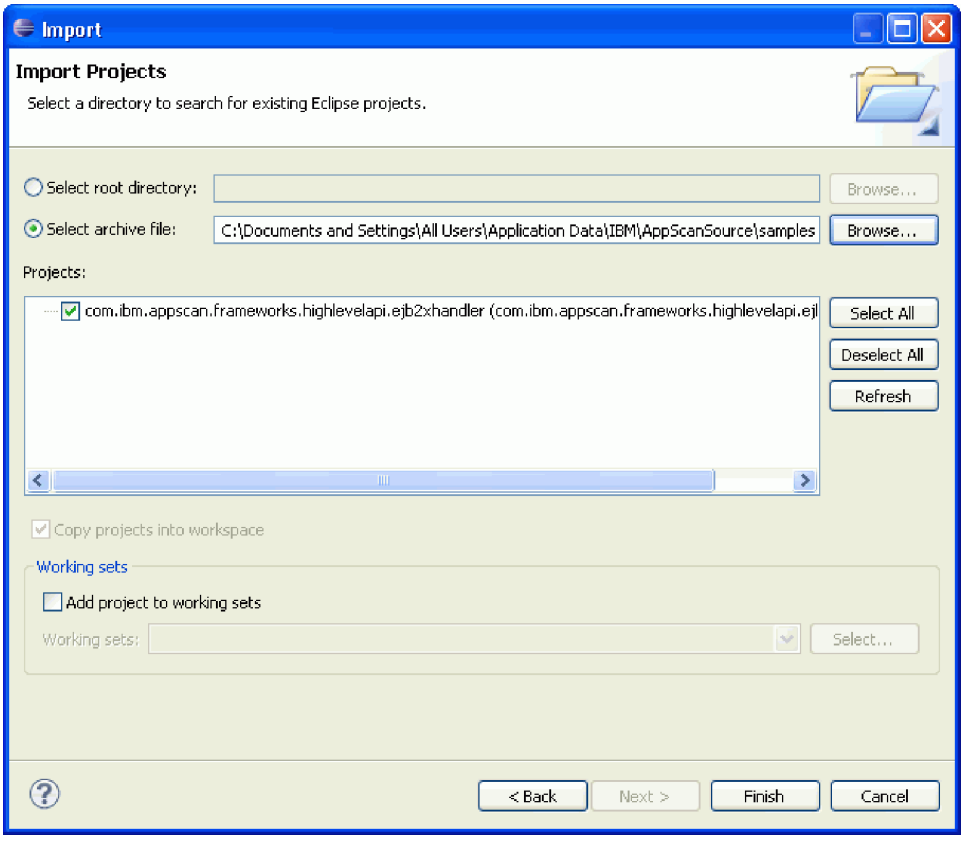
4. Dans la page **Sélectionner** de l'assistant d'importation, sélectionnez **Projets existants dans l'espace de travail** et cliquez sur **Suivant**.



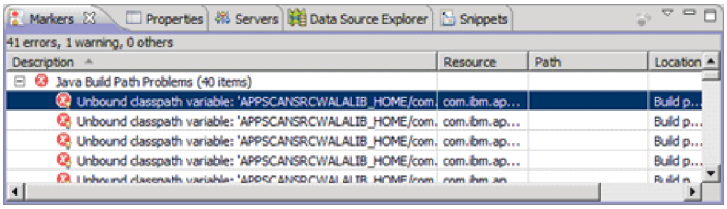
5. Sélectionnez le bouton d'option **Sélection d'un fichier archive** et cliquez sur **Parcourir**. Dans la boîte de dialogue **Sélection de l'archive contenant les projets à importer**, localisez le fichier `<data_dir>\samples\F4FEjbExample.zip` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151) et cliquez sur **Ouvrir**.

   Ce fichier zip est une archive Eclipse qui contient les informations de configuration et le code source de l'exemple de gestionnaire de frameworks `Ejb2xHandler`.

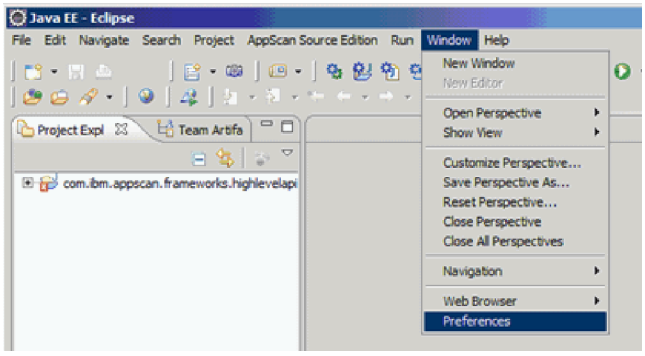
   Cliquez sur **Terminer** pour importer l'archive.



6. Une fois l'archive importée, vous constaterez la présence de plusieurs erreurs de génération. L'exemple utilise en effet une variable de chemin de classes (non encore définie à ce stade) qui pointe sur le répertoire dans lequel sont stockées ses bibliothèques.

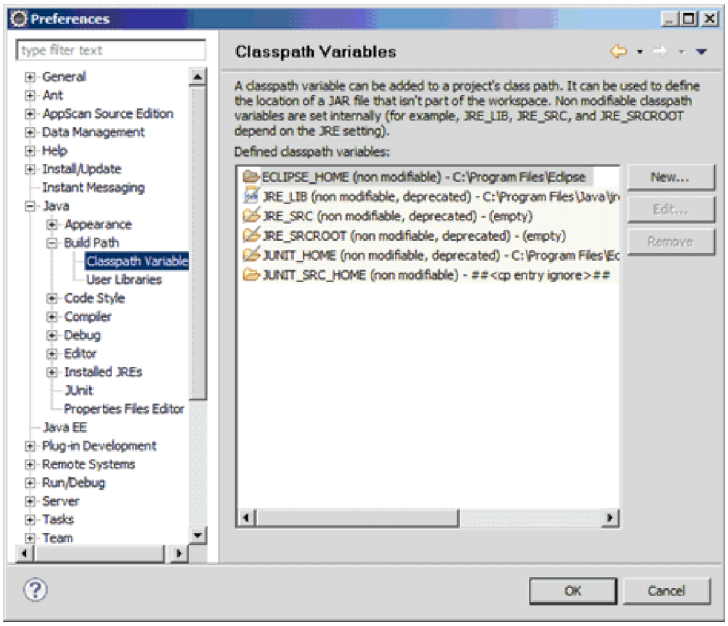


7. Vous allez maintenant définir une variable. Sélectionnez **Fenêtre** > **Préférences** sur la barre de menus principale du plan de travail.

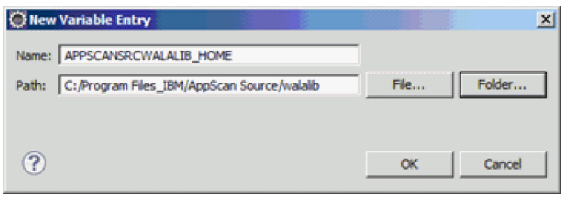


8. Dans la boîte de dialogue Préférences, sélectionnez **Java** > **Chemin de génération** > **Variables de chemin d'accès aux classes** pour ouvrir la page de

préférences correspondante. Cliquez sur **Nouveau**.



9. Créez une variable nommée `APPSCANSRCWALALIB_HOME` et, dans la zone **Chemin**, spécifiez `<install_dir>\walalib` (où `<rép_install>` représente l'emplacement de votre installation AppScan Source).



Une fois que vous avez fermé toutes les boîtes de dialogue en cliquant à chaque fois sur **OK**, l'exemple doit être généré sans erreur.

Création d'une classe qui implémente `F4FHandler`

Pour créer un nouveau gestionnaire de frameworks, vous devez d'abord créer une classe qui implémente `F4FHandler`. Deux méthodes doivent être implémentées pour prendre en charge la fonctionnalité Framework for Frameworks.

Redéfinir `isApplicable`

But : AppScan Source appelle `isApplicable` pour déterminer s'il doit ou non exécuter votre gestionnaire. Si vous retournez `True`, il exécute votre gestionnaire en appelant `handleApp`. Si vous retournez `False`, rien d'autre n'est appelé.

Remarque : `isApplicable` inclut un test au début de la méthode pour vérifier que l'application analysée est bien une application Java. Ce n'est qu'à cette condition que la méthode poursuit son exécution.

Observez l'exemple : Dans la classe `Ejb2xHandler`, la méthode `isApplicable` vérifie d'abord si le langage est approprié (car EJB est uniquement présent dans les applications Java). Si l'application est basée sur Java, `isApplicable` recherche alors toute instance de `ejb-jar.xml`, fichier de configuration obligatoirement présent dans le cas d'une application EJB 2. Si des fichiers de configuration sont trouvés, ils

sont lus par le gestionnaire et `True` est renvoyé à AppScan Source pour lui indiquer qu'il doit appeler `handleApp` afin de traiter les informations contenues dans les fichiers de configuration.

Redéfinir `handleApp`

But : AppScan Source appelle `handleApp` pour vous permettre de déterminer et définir les informations à propos du ou des frameworks utilisés dans l'application analysée. Les paramètres `F4FApp` et `F4FAction` servent à obtenir des informations à propos de l'application et à définir les modalités de gestion des frameworks présents dans cette application (et que votre gestionnaire doit prendre en charge).

Création d'un fichier manifeste pour votre gestionnaire

Créez un fichier nommé `Manifest.txt`, contenant les informations à incorporer dans le manifeste de votre fichier JAR lorsqu'il sera exporté. Plus précisément, lors du chargement de l'archive JAR de votre gestionnaire, AppScan Source recherchera l'entrée `Framework-Handler` dans le manifeste pour déterminer quelle classe contient la fonctionnalité de `F4FHandler`.

Placez les deux lignes suivantes dans le fichier `Manifest.txt` :

```
Manifest-Version: 1.0
Framework-Handler: package.HandlerClassName
```

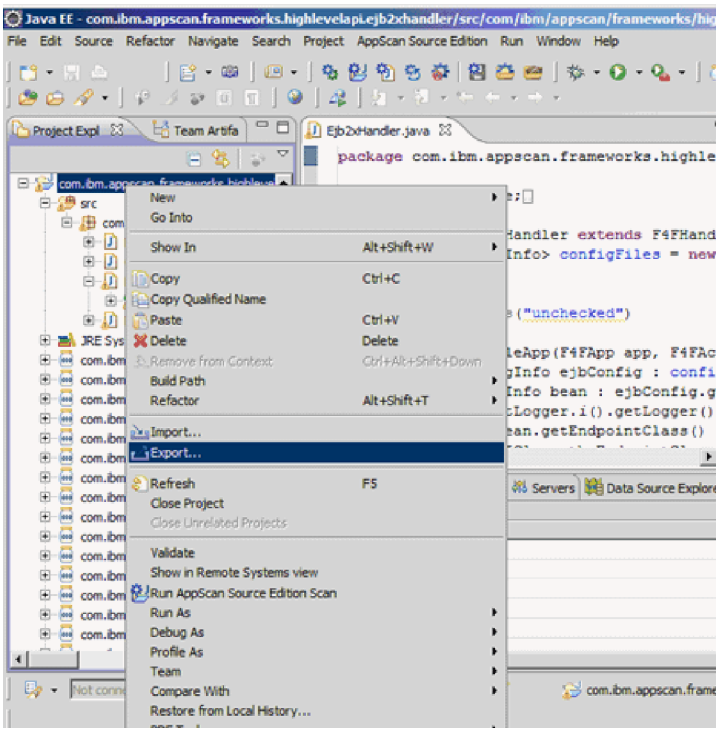
Où `package.HandlerClassName` représente le nom complet, package compris, de la classe qui implémente l'interface `F4FHandler`.

Observez l'exemple : `Manifest.txt` contient des informations qui seront placées dans le manifeste de l'archive JAR de `Ejb2xHandler` lors de sa création. Il n'y a que deux lignes, une qui indique la version et l'autre qui spécifie la classe de gestion des frameworks, `Framework-Handler: com.ibm.appscan.frameworks.highlevelapi.ejb2xhandler.Ejb2xHandler`.

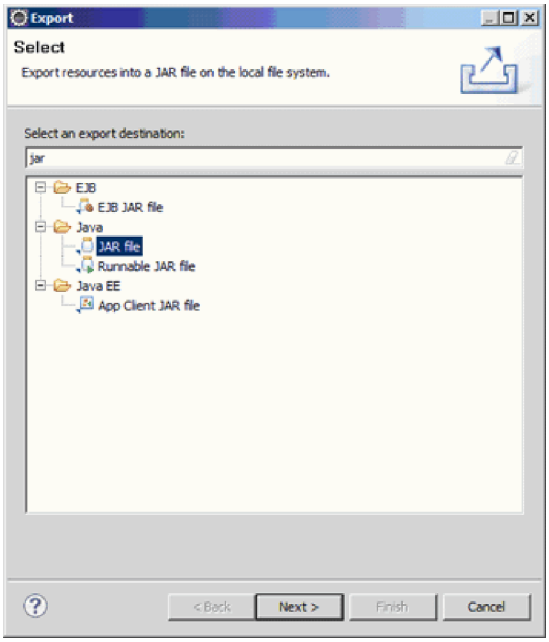
Création d'un fichier JAR pour votre gestionnaire

Procédure

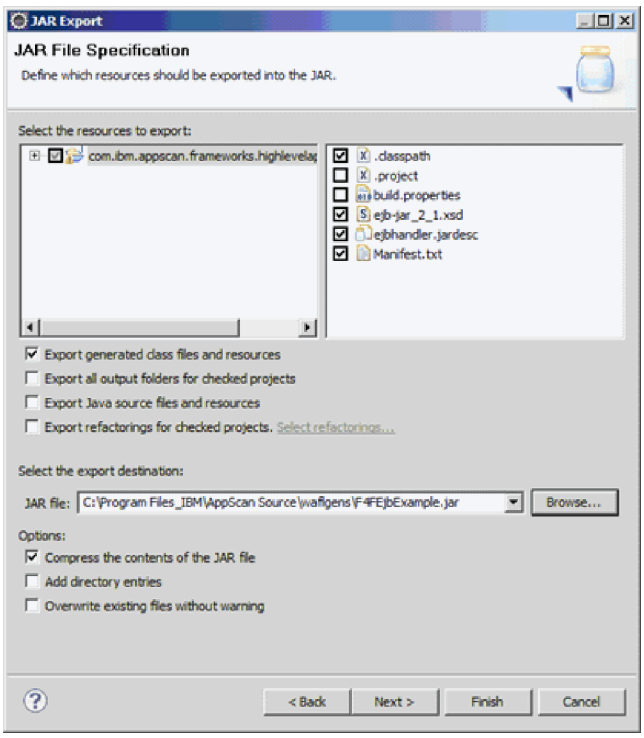
1. Dans la vue Explorateur de projets du plan de travail Eclipse, faites un clic droit sur votre projet et sélectionnez **Exporter**.



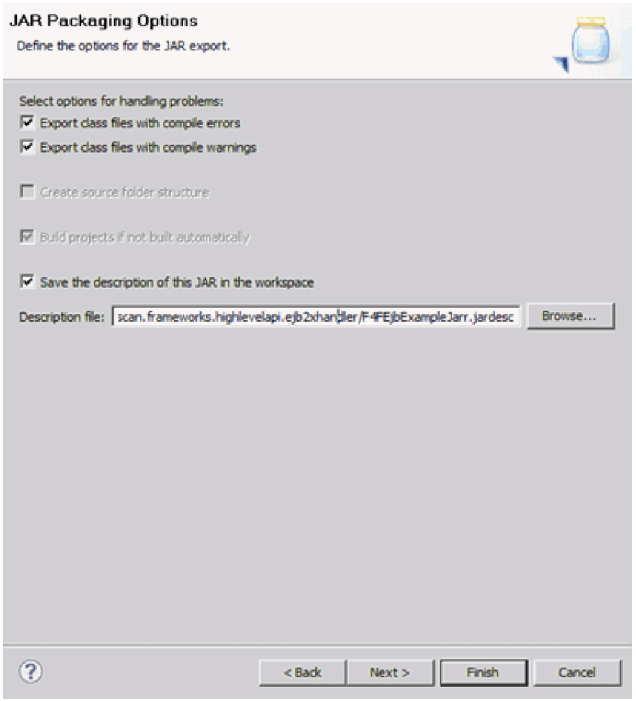
2. Dans l'assistant d'exportation, sélectionnez **Fichier JAR** et cliquez sur **Suivant**.



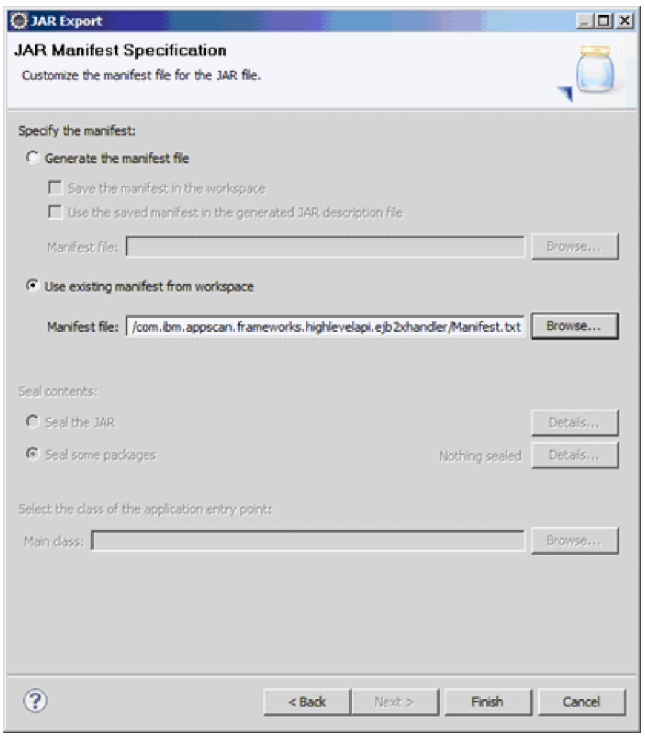
3. Dans la section **Sélectionnez la destination de l'exportation** de la boîte de dialogue Exportation de fichier JAR, sélectionnez un emplacement approprié pour votre fichier JAR et indiquez un nom. Cliquez sur **Suivant**.



4. Cliquez à nouveau sur **Suivant**. Au besoin, vous pouvez également définir l'emplacement et le nom d'un fichier de description. Un fichier `.jardesc` sera alors sauvegardé dans votre projet et contiendra tous les paramètres d'exportation de fichier JAR définis dans le cadre de la présente exportation. Ce fichier vous permettra de répéter l'exportation sans avoir à spécifier à nouveau les paramètres.



5. Sélectionnez le bouton d'option **Utiliser le manifeste existant de l'espace de travail** et cliquez sur **Parcourir**. Sélectionnez le fichier `Manifest.txt` que vous avez créé plus tôt.



6. Cliquez sur **Terminer**.

Exportation du fichier JAR vers `waflgens`

Si vous n'avez pas exporté directement le fichier JAR du gestionnaire vers le répertoire `waflgens` lors de sa création, copiez-le maintenant. Le chemin complet est `<install_dir>\waflgens` (où `<rép_install>` représente l'emplacement de votre installation AppScan Source).

Actions communes exécutées par le gestionnaire

Créer un point d'entrée de service Web

De nombreux frameworks fournissent leurs propres points d'entrée dans une application. Un exemple courant est l'exposition de services Web qui sont soit identifiés dans un fichier de configuration, soit déclarés par des annotations à même le code. Après recherche de points d'entrée désignés dans les fichiers de configuration ou directement dans le bytecode de l'application, la méthode `F4FAction.addTaintedCallback` peut être utilisée pour créer un point d'entrée de données entachées dans la méthode appropriée.

Observez l'exemple : Dans EJB 2, les points d'entrée du service Web sont déclarés en définissant des noeuds finaux (*endpoints*) dans le fichier de configuration de l'application (`ejb-jar.xml`). `handleApp` effectue alors une boucle via les beans déclarés dans `ejb-jar.xml` et chaque fois qu'une classe de noeud final est définie, il obtient la liste des noms de méthode. Il déclare ensuite leurs implémentations en tant que points d'entrée de service Web à l'aide de la méthode `addTaintedCallback`.

Remplacer une méthode

Les frameworks modernes font un usage fréquent de fonctions virtuelles et d'abstraction pour coupler plus lâchement les composants métier. Ce procédé tend à améliorer le processus de développement, mais il rend plus difficile l'analyse statique lorsque la connexion entre une fonction virtuelle et son implémentation est gérée dans un fichier de configuration ou via des annotations à même le code. `F4FAction.replaceCalls` permet à un gestionnaire de désigner ces connexions.

Observez l'exemple : Dans EJB 2, chaque bean possède un ensemble d'interfaces (locales et distantes) qui déclarent la façon dont les autres beans interagissent avec lui. Cela signifie que, chaque fois qu'une interface de bean `class.method` est appelée, elle est remplacée par le framework associé à la méthode `ImplementationClass.method` réelle.

A partir de la ligne 62, notre exemple de gestionnaire parcourt chaque bean en boucle, prend ses interfaces distantes et locales, puis les remplace par leurs implémentations réelles.

Consignation

Un gestionnaire peut utiliser la classe `com.ibm.wala.andromeda.util.logging.TaintLogger` pour consigner des messages d'information pendant l'exécution et pour provoquer l'affichage de messages d'erreur dans l'interface utilisateur AppScan Source. La classe `TaintLogger` utilise la bibliothèque `log4j`. Pour consigner un message, obtenez d'abord un objet `Logger` en appelant `TaintLogger.i().getLogger()`. Appelez ensuite les méthodes de consignation sur l'objet `Logger` (par exemple, `Logger.warn`) afin de consigner les messages souhaités. Les messages de journal apparaissent dans `<data_dir>\logs\scanner_exceptions.log` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151). Si vous utilisez `Logger.error` ou `Logger.fatal` pour consigner un message, le message d'erreur s'affichera également dans la vue Console de l'interface utilisateur AppScan Source.

Framework for Frameworks API classes and methods

This section contains the Framework for Frameworks API classes and methods, abbreviated for Adobe PDF. To access the complete set of API documentation, launch the AppScan Source for Analysis online help and navigate to Reference > Framework for Frameworks handling APIs > Framework for Frameworks API classes and methods.

The complete set of API documentation can also be found at http://www.ibm.com/support/knowledgecenter/SSS9LM/welcome.

F4FActions

```
java.lang.Object
  extended by com.ibm.appscan.frameworks.highlevelapi.F4FActions

public class F4FActions
extends java.lang.Object
```

Class for specifying how the application's framework constructs should be modeled. An `F4FHandler` mutates the `F4FAction` object passed to `F4FHandler.handleApp(F4FApp, F4FActions)` as it analyzes the application.

Constructor Detail

`F4FActions`

```
public F4FActions()
```

Create an empty `F4FActions` object. Should not be needed for implementing a new framework handler, as the relevant `F4FActions` object will be passed to `F4FHandler.handleApp(F4FApp, F4FActions)`.

addTaintedCallback

```
public void addTaintedCallback(IMethod method,
                               int numParams)
```

Same as «addTaintedCallback» `(String, int)`, but takes an `IMethod` directly rather than a VDB signature

addTaintedCallback

```
public void addTaintedCallback(java.lang.String vdbMethodSig,
                               int numParams)
```

Make a method a tainted callback, with all parameters tainted.

Remarque : For .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`.

Parameters:

- `vdbMethodSig` - the signature of the `callback` method
- `numParams` - the number of parameters for the `callback` method, including the `this` parameter

replaceCalls

```
public void replaceCalls(java.lang.String oldVDBSig,
                         java.lang.String newVDBSig)
```

Replace all calls to one method with calls to another method. We require that the descriptors for the old and new method (i.e., the number of arguments, argument type, and return type) are identical.

Remarque : replacement will only occur when `oldVDBSig` is the _declared_ target at a call site. So, if `oldVDBSig` is `Integer.toString()`, and we see a call to `Object.toString()`, we will _not_ perform a replacement at that call site, even though it might invoke `Integer.toString()`.

Remarque : for .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`

Parameters:

- `oldVDBSig` - signature of method whose calls should be replaced
- `newVDBSig` - signature of method to replace calls with

replaceCallsWithSyntheticExpr

```
public void replaceCallsWithSyntheticExpr(java.lang.String vdbSig,
    com.ibm.appscan.frameworks.specinfo.SyntheticExpr expr)
```

Replace all calls to a method with an arbitrary WAFL `SyntheticExpr`. For example, one could replace calls with an assignment via an `AssignmentExpr`.

Remarque : replacement will only occur when `oldVDBSig` is the _declared_ target at a call site. So, if `oldVDBSig` is `Integer.toString()`, and we see a call to `Object.toString()`, we will _not_ perform a replacement at that call site, even though it might invoke `Integer.toString()`.

Remarque : for .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`

Parameters:

- `vdbSig` - signature of method whose calls should be replaced
- `expr` - synthetic expression to replace calls with

replaceCallsWithParamPattern

```
public void replaceCallsWithParamPattern(java.lang.String oldVDBSig,
                                         java.util.Map<java.lang.String,
                                         java.util.Map<java.lang.Integer,
                                         java.util.regex.Pattern>>
                                         newSig2Pattern)
```

Replace calls to one method with calls to another method only if the parameters of `String` type are constants meeting specified patterns. We require that the descriptors for the old and new method (i.e., the number of arguments, argument type, and return type) are identical.

Remarque : replacement will only occur when `oldVDBSig` is the _declared_ target at a call site. So, if `oldVDBSig` is `Integer.toString()`, and we see a call to `Object.toString()`, we will _not_ perform a replacement at that call site, even though it might invoke `Integer.toString()`.

Remarque : for .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`

Parameters:

- `oldVDBSig` - signature of method whose calls should be replaced

- `newSig2Pattern` - maps VDB signature of each possible replacement method `m` to a map `M` from integer parameter positions to `Patterns`. If the string constant parameters in the appropriate positions match the patterns in `M` at some call site, a replacement to `m` will be performed.

addFrameworkInfo

```
public void addFrameworkInfo
(com.ibm.appscan.frameworks.specinfo.IFrameworkInfo info)
```

Add arbitrary additional framework info. This method should only be needed for rare cases where the other APIs provided are insufficient.

addTaintedCallback

```
public void addTaintedCallback(java.lang.String vdbMethodSig,
                               java.util.Collection<TaintedParam>
                               taintedParams)
```

Make some method a tainted callback, with only certain parameter access paths being treated as tainted.

Remarque : for .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`

Parameters:

- `vdbMethodSig` - the signature of the callback method, in VDB format
- `taintedParams` - information on which parameter access paths should be tainted

addHighLevelSyntheticMethod

```
public void addHighLevelSyntheticMethod(HighLevelSyntheticMethod m)
```

equivalent to `addHighLevelSyntheticMethod(m, true)`

addHighLevelSyntheticMethod

```
public void addHighLevelSyntheticMethod(HighLevelSyntheticMethod m,
                                        boolean isEntrypoint)
```

Add a high-level synthetic method. A corresponding WAFL synthetic method (possibly an entrypoint) will be generated.

Parameters:

- `m` - the method
- `isEntrypoint` - should the method be marked as an entrypoint in WAFL?

createGlobal

```
public Global createGlobal(java.lang.String name,
                           java.lang.String declaredVDBType,
                           boolean isEntrypointScoped)
```

Create a new global that can be accessed from `HighLevelSyntheticMethods`.

Parameters:

- `name` - name for the global
- `declaredVDBType` - the declared type of the global (e.g., `java.lang.String`).

Remarque : for .NET apps, we need a fully-qualified VDB type. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`

- `isEntrypointScoped` - if true, the global is scoped to a single entrypoint (i.e., it is request-scoped). Otherwise, the global is scoped across entrypoints (i.e., it is "session" or "application" scoped)

Returns:

- a Global object, which can be read/written inside a `HighLevelSyntheticMethod`

createGlobal

```
public Global createGlobal(java.lang.String name,
                           IClass declaredClass,
                           boolean isEntrypointScoped)
```

Just like «createGlobal», à la page 124(`String, String, boolean`), but takes an `IClass` for the declared type instead of a type name

getGlobals

```
public java.util.Collection<Global> getGlobals()
```

For internal usage.

getAdditionalFrameworkInfo

```
public java.util.Collection
<com.ibm.appscan.frameworks.specinfo.IFrameworkInfo>
getAdditionalFrameworkInfo()
```

For internal usage.

getCallReplacement2SigsInfo

```
public java.util.Map
<java.lang.String,java.util.Map
<java.lang.String,java.util.Map
<java.lang.Integer,java.util.regex.Pattern>>>
getCallReplacement2SigsInfo()
```

For internal usage.

getCallReplacement2ExprInfo

```
public java.util.Map
<java.lang.String,com.ibm.appscan.frameworks.specinfo.SyntheticExpr>
getCallReplacement2ExprInfo()
```

For internal usage.

getCallback2TaintedParams

```
public java.util.Map
<java.lang.String,java.util.Collection<TaintedParam>>
getCallback2TaintedParams()
```

For internal usage.

getHighLevelSyntheticMethods

```
public java.util.List
<com.ibm.wala.util.collections.Pair
<HighLevelSyntheticMethod,java.lang.Boolean>>
getHighLevelSyntheticMethods()
```

For internal usage.

toString

```
public java.lang.String toString()
```

Overrides:

- `toString` in class `java.lang.Object`

F4FApp

```
java.lang.Object
  extended by com.ibm.appscan.frameworks.highlevelapi.F4FApp

public class F4FApp
extends java.lang.Object
```

Representation of an application, with methods to query various properties of classes, methods, etc. Implemented mostly by delegating to methods from the T.J. Watson Libraries for Analysis (WALA); the goal is to consolidate the most useful WALA methods in a single type. See the WALA home page (http://wala.sourceforge.net) for full details on WALA APIs.

Constructor Detail

```
public F4FApp(IClassHierarchy cha)
```

Should not be needed to implement a new handler. The relevant `F4FApp` object will be passed as a parameter to `F4FHandler.handleApp(F4FApp, F4FActions)`

getAppClass

```
@Deprecated
public IClass getAppClass(java.lang.String vdbClassName)
```

Deprecated. Use `getIClass(String)` instead; this method simply delegates to that one.

getIClass

```
public IClass getIClass(java.lang.String vdbClassName)
```

get the `IClass` for some class in the application, including library jars/DLLs. If no class with the provided name is found, return `null`

Parameters:

- `vdbClassName` - class name in VDB format, e.g., `java.lang.String`

getClassAnnotations

```
public java.util.Collection<Annotation>
getClassAnnotations(IClass klass)
```

Get the annotations/attributes for a class. For .NET, the result will include inherited attributes.

Parameters:

- `klass` - the class whose annotations are desired

getMethodAnnotations

```
public java.util.Collection<Annotation>
getMethodAnnotations(IMethod method)
```

Get the annotations / attributes for a method. For .NET, these will include inherited attributes.

Parameters:

- `method` - the method whose annotations are desired

getFieldAnnotations

```
public java.util.Collection<Annotation>
getFieldAnnotations(IField field)
```

Get the annotations / attributes for a field.

Parameters:

- `field` - the field whose annotations are desired

getMethodParametersAnnotations

```
public java.util.Collection<Annotation>[]
getMethodParametersAnnotations(IMethod method)
```

Get annotations on parameters as an array of Collections, where each array element gives the annotations on the corresponding parameter. Note that the `this` parameter for an instance method cannot have annotations.

Parameters:

- `method` - the method whose parameter annotations are desired

getAllApplicationClasses

```
public java.util.Collection<IClass>
getAllApplicationClasses()
```

Get all the classes in the application (i.e., excluding those in library jars).

getClassHierarchy

```
public IClassHierarchy getClassHierarchy()
```

Get the WALA class hierarchy for the application. Most handlers should be able to work via the other methods in this class and shouldn't need to operate directly on the class hierarchy. But, access is provided for advanced use.

getMethodsDeclaredInClass

```
public java.util.Collection<IMethod>
getMethodsDeclaredInClass(IClass klass)
```

Get all the static and instance methods declared in `klass`

getClassMethods

```
public java.util.Collection<IMethod>
getClassMethods(java.lang.String className,
                java.lang.String methodName)
```

Get all the methods in an class with a particular name. If the class cannot be found, returns an empty Collection.

Parameters:

- `className` - the class name in VDB (i.e., source-level) format, e.g., `java.lang.String`
- `methodName` -

getClassMethods

```
public java.util.Collection<IMethod>
getClassMethods(IClass appClass,
                java.lang.String methodName)
```

Get all the methods in an class with a particular name.

Parameters:

- `appClass` - the class
- `methodName` -

getStringConstantsReturnedByMethod

```
public java.util.Collection<java.lang.String>
getStringConstantsReturnedByMethod(IMethod method)
```

Get the possible String constants returned by the method. E.g., if the method has a statement return "result";, then "result" will be in the returned Collection. Throws an `IllegalArgumentException` if the return type of method is not `String`.

F4FHandler

```
java.lang.Object
  com.ibm.appscan.frameworks.highlevelapi.F4FHandler

public abstract class F4FHandler
extends java.lang.Object
```

The abstract class that any new framework handler should extend.

Remarque : Any sub-class of `F4FHandler` must have a no-argument constructor, for instantiation using reflection

Constructor Detail

```
F4FHandler

public F4FHandler()
```

handleApp

```
public abstract void handleApp(F4FApp app,
                               F4FActions actions)
```

Define what actions should be represented in the generated WAFL specification to handle the given application.

Parameters:

- `app` - the application to be analyzed
- `actions` - the actions to be taken; implementations of `handleApp(F4FApp, F4FActions)` should mutate this parameter and store the desired actions

isApplicable

```
public abstract boolean isApplicable()
```

Is the framework handler applicable for the target application? Implementations should check the language of the app, whether relevant configuration files are present, etc.

setFrameworksInput

```
public void setFrameworksInput(FrameworksInput input)
```

Should not be invoked by a framework handler.

getFrameworksInput

```
protected FrameworksInput getFrameworksInput()
```

Get the parsed representation of the input parameters to the frameworks code.

TaintedParam

```
java.lang.Object
  extended by com.ibm.appscan.frameworks.highlevelapi.TaintedParam

public class TaintedParam
extends java.lang.Object
```

A type used to represented how some method parameter may reference tainted data.

Constructor Detail

```
public TaintedParam(int paramPos,
                    java.lang.String accessPath)
```

Creates a new `TaintedParam` object for a particular parameter position and access path.

Parameters:

- `paramPos` - the position of the tainted parameter, numbered starting from 0 (where the `this` parameter of an instance method is parameter 0)
- `accessPath` - access path of fields from the parameter that should be tainted, e.g., "f.g". Use "" to indicate the parameter itself should be tainted.

getParamPos

```
public int getParamPos()
```

getAccessPath

```
public java.lang.String getAccessPath()
```

hashCode

```
public int hashCode()
```

Overrides:

- `hashCode` in class `java.lang.Object`

equals

```
public boolean equals(java.lang.Object obj)
```

Overrides:

- `equals` in class `java.lang.Object`

toString

```
public java.lang.String toString()
```

Overrides:

- `toString` in class `java.lang.Object`

Méthodes synthétiques de haut niveau

Les méthodes synthétiques sont des constructions utiles pour la modélisation de flux de données avancés dans les frameworks (structures). Par exemple, de nombreux frameworks standard (tels que Struts et Spring) encouragent une *architecture MVC* (model-view-controller) pour l'application. Avec une structure MVC, la soumission du formulaire client est souvent gérée de la façon suivante :

1. En fonction de l'URL, déterminez la classe du *model* de l'application `M` pour contenir les données de formulaire soumises et la classe de *controller* `C` contenant la logique applicative.
2. Créez un objet de modèle `M` et définissez ses propriétés en fonction des données de formulaire (non sécurisées) dans la requête HTTP. Les propriétés sont généralement définies via des JavaBeans *setter* (méthode d'accès set) (par exemple, `setName()` ou `setAddress()`).
3. Effectuez une validation des données de l'objet `M`.
4. Créez un objet de contrôleur `C` et transmettez l'objet `M` à une méthode `C.execute()` qui exécute la logique applicative. Généralement, `execute()` renvoie le nom d'une vue pour fournir un résultat.
5. En fonction du nom de la vue, déterminez le fichier de vue approprié (par exemple, une page JavaServer) à afficher. Souvent, les données de l'objet `M` sont transmises à la vue via les attributs de l'objet de requête ou de session.

Toutes les fonctions précédemment décrites peuvent être modélisées avec les méthodes synthétiques Framework for Frameworks, cela permettant d'exposer les comportements en vue de leur analyse par AppScan Source. L'API Framework for Frameworks fournit des méthodes synthétiques de haut niveau qui facilitent la génération de méthodes synthétiques.

Remarque : Les noeuds de trace dont le nom de classe commence par `Appscan.Synthetic`, `Appscan.Synthetic.Validator` et `AppScan.Synthetic.Replacement` correspondent à des méthodes synthétisées par AppScan Source.

- Les méthodes `AppScan.Synthetic` permettent d'assembler des traces dans le code d'application qui utilisé des structures.

- Une méthode `AppScan.Synthetic.Validator` modélise la validation sous-jacente effectuée par l'exécution de la structure. Vous pouvez sélectionner une méthode valideur et la marquer comme **Validator** si nécessaire.
- Une méthode `AppScan.Synthetic.Replacement` indique qu'une méthode du code d'application a été remplacée par AppScan Source pour capturer le flux de données entre des composants disjoints (tels que les contrôleurs et les vues) de la structure.

Format VDB

Les méthodes de l'API Framework for Frameworks requièrent que les éléments de chaîne (`String`) représentant des noms de type ou des signatures de méthode soient au format VDB. Les noms de type VDB sont simplement des noms qualifiés complets de niveau source (par exemple, `java.lang.String`), ou, dans le cas de classes internes, sont représentés avec un symbole dollar ($)(par exemple `javax.swing.text.DefaultEditorKit$DefaultKeyTypedAction`). Pour `.NET`, les noms abrégés tels que `int` et `string` doivent être évités en cas d'utilisation de l'API Framework for Frameworks. Utilisez plutôt des noms qualifiés complets tels que `System.Int32` et `System.String`.

Une signature de méthode VDB se compose de deux parties :

1. Le nom de la méthode, incluant le nom qualifié complet de la classe englobante (par exemple, `java.lang.String.substring`).
2. Un descripteur pour la méthode, fournissant les types de paramètre et le type de retour. Les types de paramètres sont entre parenthèses et séparés par des points-virgule (`;`). La parenthèse fermante est suivie d'un signe deux-points (`:`), puis du type de retour.

Voici un exemple de signature de méthode VDB :

```
java.lang.String.substring(int;int):java.lang.String
```

Voici un exemple de signature de méthode VDB d'une classe interne :

```
javax.swing.text.DefaultEditorKit$DefaultKeyTypedAction.
    actionPerformed(ActionEvent):void
```

Pour la plupart des méthodes API qui requièrent un élément `String` au format VDB, il existe souvent une méthode équivalente qui utilise plutôt un objet `IClass` ou `IMethod`. Par exemple, la méthode `F4FActions.addTaintedCallback(String,int)` nécessite que son premier paramètre soit une signature de méthode au format VDB alors que `F4FActions.addTaintedCallback(IMethod,int)` identifie la méthode avec un objet `IMethod`. Il peut s'avérer plus simple d'utiliser les méthodes API utilisant des objets `IClass` et `IMethod` pour deux raisons :

1. L'utilisateur n'a plus à se soucier du formatage VDB de ces méthodes car cela est géré en interne.
2. Lorsque l'on obtient l'objet `IClass` ou `IMethod` (par exemple, via `F4FApp.getClassMethods()`), on est sûr que la méthode ou la classe correspondante existe réellement dans le code d'application.

Utilisation de méthodes synthétiques de haut niveau

Cette rubrique explique comment utiliser certaines des fonctions clés des méthodes synthétiques de haut niveau. Pour plus d'informations, consulter la documentation Javadoc relative aux méthodes et classes API Framework for Frameworks.

- «Création d'une méthode synthétique de haut niveau», à la page 132
- «Création de variables locales», à la page 132

- «Ajout d'appels»
- «Ajout d'une valeur de retour», à la page 133
- «Objets Global», à la page 133

Création d'une méthode synthétique de haut niveau

Pour créer une méthode synthétique de haut niveau et l'ajouter à l'objet `F4FActions actions`, utilisez les codes suivants :

```
HighLevelSyntheticMethod m = HighLevelSyntheticMethod.make();
actions.addHighLevelSyntheticMethod(m);
```

Vous pouvez spécifier le nom, les types de paramètre et le type de retour de la méthode synthétique en transmettant une signature de méthode VDB à `HighLevelSyntheticMethod.make()`.

Création de variables locales

Il est possible de créer un variable locale pour une méthode synthétique de haut niveau `m` en procédant comme suit :

```
Local l = m.newLocal("java.lang.String");
```

Il n'est pas nécessaire d'appeler les constructeurs dans les méthodes synthétiques. Lorsque l'on utilise le code précédemment indiqué, on peut supposer que `l` fait référence à un objet `String` non nul lors de l'ajout d'instructions supplémentaires à la méthode synthétique.

Ajout d'appels

La plupart des instructions figurant dans les méthodes synthétiques de haut niveau sont des appels de méthode ajoutés via la méthode `addCall()`. Les paramètres de `addCall()` représentent la méthode à appeler, les informations d'emplacement de fichier pour l'appel et les paramètres à transmettre lors de l'appel. Voici un exemple d'ajout d'un appel à une méthode d'accès set `sample.BeanType.setName()`, en prenant un objet `F4FApp app` :

```
Collection<IMethod> setterMethods =
   app.getClassMethods("sample.BeanType", "setName");
// assume there is exactly one "setName" method in BeanType
IMethod setter = setterMethods.iterator().next();
HighLevelSyntheticMethod m = HighLevelSyntheticMethod.make();
Local l = m.newLocal("sample.BeanType");
m.addCall(setter, null, l, Taint.taint());
```

Les étapes 2 à 5 de la gestion de la soumission du formulaire client dans une architecture MVC (comme indiqué dans «Méthodes synthétiques de haut niveau», à la page 130) peuvent toutes faire l'objet d'un certain degré de modélisation en ajoutant des appels appropriés à une méthode synthétique de la façon mentionnée précédemment.

Les paramètres d'un appel sont représentés par les objets de type `Param`, qui peuvent être l'un des suivants :

- Objet `Local`, représentant une variable locale
- Objet `Taint`, représentant des données non sûres ou *tâchées*.
- Objet `EnclosingFormal`, représentant un paramètre formel de la méthode synthétique de haut niveau. Par exemple, si vous disposez d'une méthode

synthétique avec la signature `synthMethod(int):void`, `EnclosingFormal.FIRST` fait référence au paramètre `int` de la méthode.

- Objet `Global` représentant des données globales (présenté dans «Objets Global»).

Remarque : Pour les méthodes d'instance non statiques, la valeur à transmettre sous la forme `this` doit être fournie à `addCall()`. Dans l'exemple fourni précédemment, la valeur dans `l` est transmise sous la forme `this` à `setName()`.

Ajout d'une valeur de retour

Une méthode synthétique peut renvoyer une valeur à l'aide de la méthode `setReturnedValue()`. Les valeurs de retour peuvent être utiles pour générer des méthodes de marqueur permettant de modéliser la validation de structure complexe. Par exemple, si vous utilisez une structure qui effectue la validation complexe d'une requête HTTP contaminée avant de la transmettre à la méthode d'accès set d'un objet de modèle, vous pouvez exposer la validation sur les traces découvertes par AppScan Source en utilisant un code similaire au suivant :

```
String validatorSig =
   "Synth.validatorModel(java.lang.String):
   java.lang.String";
HighLevelSyntheticMethod validator =
   HighLevelSyntheticMethod.make(validatorSig);
// just return the parameter passed in
validator.setReturnedValue(EnclosingFormal.FIRST);
HighLevelSyntheticMethod m = ...;
Local validated = m.addCall(validatorSig, null, Taint.taint());
// now, validated can be passed to a setter
```

La méthode synthétique `Synth.validatorModel()` renvoie simplement la chaîne `String` qui lui est transmise en tant que paramètre. Ensuite, un appel à `Synth.validatorModel()` est ajouté à une autre méthode synthétique, transmettant ainsi une valeur contaminée en tant qu'argument. Le résultat de cet appel est stocké dans `validated`, qui peut être transmis dans un appel ultérieur à une méthode d'accès set (comme illustré par l'exemple figurant dans «Ajout d'appels», à la page 132). Les traces impliquant ces données contaminées incluent l'appel à `Synth.validatorModel()` et un utilisateur AppScan Source peut choisir de filtrer les traces si la validation est jugée suffisante.

Objets Global

Les objets *Global* sont des fonctions avancées qui permettent d'exposer un flux entre les différentes parties d'une application. Par exemple, les objets Global peuvent être utilisés pour modéliser le flux de données d'un contrôleur en direction d'une vue via des attributs de requête ou de session. Les opérations de base permettant de créer des objets Global et d'y accéder sont les suivantes :

- Pour créer un objet `Global` représentant des données globales, utilisez `F4FActions.createGlobal()`.
- Pour écrire dans un objet Global, utilisez `HighLevelSyntheticMethod.addGlobalWrite()`.
- Pour lire un objet Global, transmettez l'objet `Global` sous forme de paramètre `Param` dans un appel envoyé à `HighLevelSyntheticMethod.addCall()` ou faites en sorte qu'il soit renvoyé par une méthode synthétique via `HighLevelSyntheticMethod.setReturnedValue()`.

Exemple : Une classe de contrôleur écrit le prénom d'un utilisateur dans l'attribut de demande `firstName`, puis la vue lit cet attribut de demande et renvoie la valeur dans la réponse. A un haut niveau, on pourrait modéliser ce flux de la manière suivante :

```
Global firstNameGlobal = actions.createGlobal
   ("firstName", "java.lang.String", true);
HighLevelSyntheticMethod controller = ...;
Local firstNameLocal = controller.newLocal("java.lang.String");
controller.addGlobalWrite(firstNameGlobal, firstNameLocal, null);
HighLevelSyntheticMethod view = ...;
view.addCall("Response.write(java.lang.String):void",
   null, firstNameGlobal);
```

En ajoutant une écriture dans `firstNameGlobal` dans la méthode synthétique du contrôleur, puis en transmettant `firstNameGlobal` à `Response.write()` dans la méthode synthétique `view`, le flux de données allant du contrôleur vers la vue est exposé.

Remarque : Cet exemple contient de nombreuses simplifications. Une version complète devrait exposer le flux de données approprié à `firstNameLocal`.

Exemple : création d'une méthode synthétique

Cet exemple illustre la création d'une méthode synthétique. Dans cette méthode synthétique, une zone de classe est d'abord rendue vulnérable aux taches, puis un appel est ajouté vers une méthode du code de l'utilisateur, et le résultat de cet appel de méthode est transmis à un collecteur.

- «Scénario d'application»
- «Composants modélisés dans la méthode synthétique»
- «Etape 1 : Créer une méthode synthétique vide», à la page 135
- «Etape 2 : Modéliser la vulnérabilité aux taches d'une zone de classe», à la page 135
- «Etape 3 : Modéliser l'appel de la méthode createUser()», à la page 136
- «Etape 4 : Modéliser le renvoi des données au client», à la page 136

Scénario d'application

Dans cet exemple, `createUser` est une API REST implémentée dans la structure JAX-RS. Un utilisateur peut appeler cette API via une adresse URL telle que `http://host:port/users/createUser`. L'environnement d'exécution de la structure délègue cet appel vers `UserRestService.createUser()` car le chemin correspond. En outre, il initialise la variable de classe `urlInfo` des données client avant d'appeler `createUser()`. Finalement, les données retournées par `createUser()` sont renvoyées au client par l'environnement d'exécution.

Composants modélisés dans la méthode synthétique

Pour la capture du «Scénario d'application» dans une méthode synthétique, les éléments suivants sont modélisés :

- la vulnérabilité aux taches de la variable de classe `urlInfo` de `UserRestService`,
- l'appel de `UserRestService.createUser()`,
- le retour des données au client.

Voici le code de la méthode `createUser()` :

```
import java.io.BufferedWriter;

@Path("users")
public class UserRestService {
  @Context
  UriInfo urlInfo;

  @Path("/createUser")
  public User createUser(){
    MultivaluedMap<String, String> queryParams =
      urlInfo.getQueryParameters();
    String name = queryParams.getFirst("name");
    User user = new User(name);
    return user;
  }
}
```

Etape 1 : Créer une méthode synthétique vide

```
22    public class JAXRSHandler extends F4FHandler{
23    @Override
        public void handleApp(F4FApp app, F4FActions actions) {
24    HighLevelSyntheticMethod synthMethod = HighLevelSyntheticMethod.make();
```

- L'objet de la méthode synthétique est initialisé à la ligne 24.

Etape 2 : Modéliser la vulnérabilité aux taches d'une zone de classe

Le fragment de code ci-dessous explique comment des zones de classe spécifiques peuvent être rendues vulnérables aux taches. Dans cet exemple, l'utilisateur souhaite rendre vulnérables aux taches les zones de classes dotées de l'annotation `@Context`.

```
27    // create a local variable of the appropriate type
28    Local userRestServiceClazzInstance =
        synthMethod.newLocal("com.ibm.appscan.UserRestService");
29
30    // get all the class fields
31    for(IField field: app.getIClass
        ("com.ibm.appscan.UserRestService").getDeclaredInstanceFields()){
32
33      //get all the annotations associated with the field
34      Collection<Annotation> fieldAnnotations = app.getFieldAnnotations(field);
35
36      //for each annotation of the field check if it is an @Context annotation
37      for (Annotation annotation : fieldAnnotations) {
38
39        if (annotation.getType().getName().toString().equals
            ("Ljavax/ws/rs/core/Context")) {
40
41          // call the F4F API to assign taint to the field
43          synthMethod.addInstanceVariableWrite(
                 userRestServiceClazzInstance /*Variable representing
                 class instance*/,
44               field /*field to taint  */,
                 Taint.taint() /* taint */,
                 null);
45        }
46      }
47    }
```

L'API Framework for Frameworks `addInstanceVariableWrite` prend quatre arguments :

1. Le premier argument est la référence de classe dont la zone doit être vulnérable aux taches. Dans l'exemple, la variable locale `userRestServiceClazzInstance` fait référence à cet argument.
2. Le deuxième argument est la zone de classe à rendre vulnérable aux taches.
3. Le troisième argument est la nouvelle valeur qui sera affectée à la zone variable de la classe. Dans l'exemple, cette variable doit être rendue vulnérable aux taches pour transmettre `Taint.taint()`.
4. Le dernier argument est `FilePositionInfo` qui a la valeur `null` dans l'exemple.

Etape 3 : Modéliser l'appel de la méthode `createUser()`

Dans cette étape, l'appel est simulé dans la méthode synthétique.

```
50    // call createUser() and store the returned value to a local variable
51    Local returnObj = synthMethod.addCall(
52      "com.ibm.appscan.UserRestService.createUser():com.ibm.appscan.User"
          /* signature of the method to be called */,
          null, /* FilePositionInfo */
53        userRestServiceClazzInstance /* Object on which the method
            will be called */);
```

L'API de haut niveau `addCall` présente trois arguments :

- Le premier argument est la signature de la méthode à appeler. Dans ce cas, `User.createUser()` doit être appelée, sa signature est donc utilisée.
- Le deuxième argument est `FilePositionInfo`. Il est transmis avec la valeur `null` car il n'est pas obligatoire pour cet exemple.
- Le dernier argument représente la liste des paramètres requis pour appeler la méthode `createUser()`.

Etant donné que `createUser()` n'admet aucun argument, le seul argument transmis à cet appel est l'objet `this`, une variable `Local` (`userRestServiceClazzInstance`) du type `User`. La méthode `createUser()` renvoie un nouvel élément `User` créé et stocké dans une nouvelle variable `Local` nommée `returnedObj` (voir ligne 51).

Etape 4 : Modéliser le renvoi des données au client

La dernière étape consiste à créer un collecteur pour modéliser le renvoi des données au client.

```
58    // create a PrintWriter object
59    Local printWriterObj = synthMethod.newLocal("java.io.PrintWriter");
60
61    // we want to call the print API on the PrintWriter object.
      // The print API takes a single argument of Object type returns void
62
63    // we create a param list of size 2. The first item in this list is always the
      //  'this' object and rest are actual method arguments.
64    Param[] printWriterObjParam =  new Param[2];
65    printWriterObjParam[0] = printWriterObj;  // The "this" object
      // the value returned by the call to the createUser method
66    printWriterObjParam[1] = returnObj;
67
68    // Now add a call to the print method
69    synthMethod.addCall("java.io.PrintWriter.print(java.lang.Object):void",
        null, printWriterObjParam);
70  }
```

- A la ligne 59, un objet `Local` est créé pour la classe `PrintWriter` dans la méthode synthétique.

- De la ligne 64 à la ligne 66, la liste des paramètres requis pour appeler la méthode `print` sur l'instance de classe `PrintWriter` est préparée (`printWriterObj`) :
  - Le premier paramètre est la référence d'objet (`printWriterObj`).
  - Le deuxième paramètre est l'argument de la méthode `print`. Il faut transmettre la valeur de renvoi stockée dans la variable Local `returnObj`.
- A la ligne 69, un appel de méthode `PrintWriter.print` est ajouté pour modéliser le renvoi des données au client.

Intégration d'un nouveau gestionnaire Framework for Frameworks de services Web au scanner personnalisé WSDL (Web Service Description Language) existant

Pourquoi et quand exécuter cette tâche

Lors de l'examen d'une application JAX-RS ou JAX-WS avec AppScan Source, vous devez vérifier que vous pouvez capturer tous les points d'entrée de service Web, de sorte que vous ne manquiez aucune vulnérabilité dans votre application. Ce document explique comment procéder. Il décrit les étapes que vous devez suivre pour intégrer un nouveau gestionnaire Framework for Frameworks (F4F) utilisé pour analyser les infrastructures de service Web non prises en charge avec le scanner personnalisé WSDL (Web Service Description Language).

Ce travail est nécessaire pour que le scanner personnalisé WSDL soit en mesure de différencier les points d'entrée de service Web qui ont déjà été reconnus par ce nouveau gestionnaire F4F et les points d'entrée de service Web qui existent dans le fichier WSDL. Avec ces informations, le scanner personnalisé WSDL peut éliminer les constatations de configuration qui ont déjà été analysées par ce nouveau gestionnaire F4F.

- «Connexion du gestionnaire F4F de services Web au scanner personnalisé WSDL»
- «Mappage de signature», à la page 139

Connexion du gestionnaire F4F de services Web au scanner personnalisé WSDL

Avant de commencer

Cette section suppose que le gestionnaire F4F principal et la génération de méthode synthétique ont déjà été développés et testés.

Procédure

1. Ajoutez une dépendance depuis votre plug-in à `com.ibm.appscan.frameworks.wsdl.util`, qui inclut des API auxiliaires pour la génération de rapports concernant les services trouvés. Elle existe également dans le dossier `walalib`.
2. Dans le gestionnaire principal (qui étend la classe `F4FHandler`), au début de la méthode `handleApp()`, ajoutez les lignes suivantes :

```
String filePath = WSDLReportingUtil.getServicesFilePath
     (getFrameworksInput().getFileLocs());
if (filePath!=null) {
WSDLReportingUtil.initXmlDocument(filePath);
}
```

Ces lignes créent un fichier temporaire qui est utilisé par le scanner personnalisé WSDL.

3. A la fin de la méthode `handleApp()`, ajoutez la ligne suivante pour vous assurer que tous les services signalés sont sauvegardés avant la fermeture du gestionnaire F4F :

   ```
   WSDLReportingUtil.saveXmlDocument();
   ```

4. Pour signaler un service, ajoutez un appel à l'une des API suivantes depuis l'API `WSDLReportingUtil` :

   - `reportService(String espaceNomCible, String nomService, String signatureMéthode)`
   - `reportService(String espaceNomCible, String nomService, String nomOpération, ArrayList<String> paramètresWsdl)`

   où :

   - `espaceNomCible` est l'espace de nom cible du service. Dans JAX-WS par exemple, il se trouve dans les annotations ou correspond au nom de package.
   - `nomService` est le nom du service. Il se trouve dans les annotations ou peut correspondre au nom de classe du service.
   - `signatureMéthode` est la signature de la méthode provenant du fichier WSDL, simplifiée.
   - `nomOpération` est le nom de la méthode, qui est lié à un nom d'opération dans un type de port du fichier WSDL.
   - `paramètresWsdl` est la liste des types de paramètre au format WSDL simple.

5. Actuellement, le scanner personnalisé WSDL fonctionne de façon autonome ou dans le cadre d'un examen Java/JSP. Si le nouveau gestionnaire F4F de services Web est conçu pour fonctionner avec un langage autre que Java (par exemple .NET), vérifiez que le scanner personnalisé WSDL est appelé avec le type d'examen approprié en effectuant les opérations suivantes :

   a. Accédez au répertoire `<data_dir>/ltd` (où `<rép_données>` est l'emplacement de vos données de programme AppScan Source, comme décrit dans Chapitre 9, «Installation et emplacements des fichiers de données utilisateur», à la page 151).
   b. Effectuez une copie de sauvegarde de tous les fichiers, en cas de problème.
   c. Ouvrez le fichier `.ltd` correspondant au type d'examen approprié. Par exemple, pour C++, ouvrez le fichier `cpp.ltd`.
   d. Ajoutez la ligne suivante à la balise `<LanguageTypeDefinition>` fermante :

      ```
      <Scanner name="wsdl"/>
      ```

   e. Copiez la valeur `file_extention_set_name`, qui est requise à l'étape suivante.
   f. Ouvrez le fichier `file_extensions.xml` et localisez la chaîne `FileExtensionSet` associée au nom qui a été copié à l'étape précédente.
   g. Ajoutez la ligne suivante à cet endroit dans le fichier :

      ```
      <FileExtension extension="wsdl" assess="true"/>
      ```

   h. Sauvegardez tous les fichiers et redémarrez AppScan Source. Le scanner personnalisé WSDL s'exécutera désormais dans le cadre de l'examen d'application complet.

Résultats

A ce stade, le scanner personnalisé WSDL n'affiche que les constatations de configuration (dans la vue Constatations) des fichiers WSDL qui ne sont pas implémentés par votre infrastructure. Les services provenant de fichiers WSDL qui

sont déjà implémentés dans l'application n'apparaissent pas parmi les constatations de configuration. Ils sont analysés par le nouveau gestionnaire F4F de services Web.

Mappage de signature

Pourquoi et quand exécuter cette tâche

Le scanner personnalisé WSDL tente de mettre en correspondance les méthodes signalées avec les services WSDL, les types de port WSDL et les opérations WSDL appropriés. C'est la raison pour laquelle les services Web sont indiqués au format WSDL. Certaines simplifications ont été introduites pour une meilleure compréhension :

- Pour les types primitifs XSD, il n'est pas nécessaire d'ajouter un espace de nom cible. A la place, utilisez simplement le nom (par exemple `string`, `int`, `float`, `double`).
- Pour les collections et les tableaux, `[]` est ajouté après le type (par exemple `string[]`, `int[]`).
- Pour les types de classe qui seront mappés à des types complexes XSD (avec JAX-B ou d'autres technologies), le type apparaît comme suit :

  ```
  http//my-types/myxsdnamespace/:Customer
  ```

  L'espace de nom doit être disponible dans l'annotation Java. Si tel n'est pas le cas, il doit correspondre à l'espace de nom du service appelant ou au nom de package, selon l'implémentation de l'infrastructure.

Remarque : Les erreurs de mappage peuvent empêcher le scanner personnalisé WSDL de mettre en correspondance votre API de service Web signalée avec l'opération WSDL associée. Si tel est le cas, une constatation de configuration supplémentaire apparaît. Elle doit être considérée comme un faux positif et indique un problème de mappage.

Si vous ne savez pas à quoi doit ressembler la signature attendue, exécutez depuis AppScan Source un examen WSDL des fichiers WSDL eux-mêmes. Cet examen génère des constatations pour toutes les opérations figurant sous les services WSDL. Cliquez sur chaque constatation pour afficher la signature de méthode dans la vue Constatation détaillée.

Chapitre 8. Messages d'erreur du composant client AppScan Source

CRWSA0900E Le groupement exclu n'a pas pu être renommé

Le groupement exclus est un groupement intégré qui contient toutes les constatations exclues. Il ne peut pas être renommé.

CRWSA0907E Echec de l'ajout de <> car ce dernier existe déjà dans la base de données et ne peut pas être écrasé.

La règle personnalisée existe déjà dans la base de données AppScan Source et ne peut être ni recréée, ni écrasée. Cette erreur se produit dans les cas suivants :

- La règle envoyée dans la base de données est marquée comme un collecteur et vous tentez de créer une règle personnalisée qui marque le collecteur comme une propagation de tache.
- La règle envoyée dans la base de données est marquée comme un propagateur de tache et vous tentez de créer une règle personnalisée qui modifie cette règle en ajoutant une propagation de tache.

CRWSA0920E Des erreurs sont survenues lors de l'importation des signatures dans le projet

Vérifiez que le projet est configuré correctement avant d'importer les signatures.

CRWSA0925E La valeur entrée pour "{0}" n'est pas valide.

La valeur entrée n'est dans un format valide. Entrez un format valide.

CRWSA0930E Le format de date entré pour "{0}" n'est pas valide.

CRWSA0935E Le format de nombre entré pour "{0}" n'est pas valide.

CRWSA0940E Utilisateur inconnu

Vérifiez que l'utilisateur existe sur le serveur Enterprise Server et que vous avez entré les informations utilisateur correctement.

CRWSA0945E L'ID utilisateur spécifié est trop court. Les ID utilisateur doivent comporter au moins un caractère.

CRWSA0950E Le mot de passe spécifié est trop court. Les mots de passe doivent comporter au moins 6 caractères.

CRWSA0955E L'ID utilisateur spécifié est trop long. Les ID utilisateur ne doivent pas comporter plus de 255 caractères.

CRWSA0960E L'ID utilisateur spécifié contient des caractères non valides.

CRWSA0965E Le mot de passe spécifié est trop long. Ces mots de passe ne doivent pas comporter plus de 16 caractères.

CRWSA0970E Les mots de passe spécifiés ne concordent pas. Entrez à nouveau les mots de passe.

CRWSA0975E L'utilisateur Admin ne peut pas être modifié.

CRWSA0980E L'adresse électronique spécifiée n'est pas valide. - Le format requis est le suivant : <utilisateur>@<domaine>, où <domaine> ne peut dépasser 255 caractères.

CRWSA0985E La zone Adresse électronique est obligatoire.

CRWSA0990E La zone Nom est obligatoire.

CRWSA0995E L'ID utilisateur spécifié figure dans le référentiel, les zones Nom et Adresse électronique ont été mises à jour.

CRWSA1010W Impossible d'ajouter l'utilisateur {0}.

Pour plus d'informations, voir l'aide du serveur AppScan Enterprise Server.

CRWSA1050E Erreur inattendue lors du chargement de l'éditeur externe.

Vérifiez que l'éditeur externe est disponible et qu'il fonctionne correctement.

CRWSA1055E Erreur lors du chargement de l'éditeur externe. Le fichier de propriétés est endommagé.

Vérifiez que l'éditeur externe est disponible et qu'il fonctionne correctement.

CRWSA1060E Erreur lors du chargement de l'éditeur externe. Fichier de propriétés manquant.

Vérifiez que l'éditeur externe est disponible et qu'il fonctionne correctement.

CRWSA1065E Erreur lors du chargement de l'éditeur externe. Impossible de charger l'éditeur souhaité

Vérifiez que l'éditeur externe est disponible et qu'il fonctionne correctement.

CRWSA1070E Erreur lors du chargement de l'éditeur externe. Le plug-in de l'éditeur a renvoyé un échec.

Vérifiez que l'éditeur externe est disponible et qu'il fonctionne correctement.

CRWSA1075E Erreur lors du chargement de l'éditeur externe. Impossible de créer une instance exécutable.

Vérifiez que l'éditeur externe est disponible et qu'il fonctionne correctement.

CRWSA1080E Erreur lors du chargement d'Eclipse. Echec de la connexion.

Vérifiez que vous avez correctement configuré Eclipse afin qu'il soit utilisé comme un éditeur externe (utilisez un environnement Eclipse qui contient `com.ouncelabs.core.filelauncher.jar` dans son répertoire `plugins\`).

CRWSA1085E Erreur au chargement d'Eclipse. Pas de réponse du serveur.

Vérifiez que vous avez correctement configuré Eclipse afin qu'il soit utilisé comme un éditeur externe (utilisez un environnement Eclipse qui contient `com.ouncelabs.core.filelauncher.jar` dans son répertoire `plugins\`).

CRWSA1090E Erreur au chargement d'Eclipse. Echec du protocole.

CRWSA1095E Erreur au chargement d'Eclipse. Chemin Eclipse non valide.

Vérifiez que vous avez correctement configuré Eclipse afin qu'il soit utilisé comme un éditeur externe (utilisez un environnement Eclipse qui contient `com.ouncelabs.core.filelauncher.jar` dans son répertoire `plugins\`).

CRWSA1100E Erreur au chargement d'Eclipse. Impossible d'exécuter {0}

CRWSA1120E Echec de l'examen

CRWSA1125E Une erreur s'est produite lors de l'extraction des résultats de l'examen

CRWSA1150W Avertissement : La désignation de "{0}" en tant que routine de validation sera sans effet sur la constatation ou le graphe de trace en cours. Etes-vous certain de vouloir ajouter le noeud racine comme routine de validation ?

CRWSA1155E Au moins une source, un collecteur, une propriété de source ou une propriété de collecteur est requis.

CRWSA1160E Impossible de lancer l'assistant de règles personnalisées alors qu'une évaluation est en cours.

CRWSA1165E Erreur lors de l'ajout des règles personnalisées

CRWSA1170E Echec de l'ajout de {0}.

CRWSA1175E Echec de l'ajout de {0} car ce dernier existe déjà dans la base de données et ne peut pas être écrasé.

CRWSA1185E Echec de la validation en raison d'une erreur de compilation

CRWSA1190E Echec de la compilation

CRWSA1195E Une erreur s'est produite lors de la tentative d'ouverture de l'éditeur.

En cas d'échec de l'ouverture de l'éditeur, essayez de relancer AppScan Source.

CRWSA1200E Le fichier sélectionné est introuvable sur ce système.

CRWSA1205E Une erreur s'est produite lors de la création de marqueurs de code source pour cette évaluation.

CRWSA1210E Un ou plusieurs fichiers n'ont pas pu être localisés pour la fonction du fragment de code source. Désirez-vous être invité à fournir l'emplacement des fichiers manquants ?

CRWSA1215E Echec de la tentative de connexion à AppScan Enterprise Console. - Vérifiez que le serveur est en cours d'exécution à l'emplacement spécifié.

CRWSA1220E Expiration du délai de connexion après {0,number} secondes - Nous vous recommandons de redémarrer les services AppScan Enterprise Console.

CRWSA1225E Erreur lors du stockage des filtres non sauvegardés. Les modifications des filtres ne seront pas sauvegardées.

CRWSA1230E Impossible de sauvegarder une évaluation lorsqu'un examen est en cours.

CRWSA1240E Application introuvable

CRWSA1245E Impossible de trouver {0}

Vérifiez l'emplacement du JRE spécifié.

CRWSA1250E Impossible de localiser le plug-in startup.jar ou org.eclipse.equinox.launcher

Vérifiez l'installation Eclipse spécifiée.

CRWSA1290W Un kit JDK est manquant dans un ou plusieurs projets. Le kit JDK par défaut a été appliqué à ces projets.

Le kit JDK spécifié pour un projet importé n'a pas été reconnu par AppScan Source. Pour corriger cette erreur, ajoutez-le en tant que kit JDK reconnu à l'aide de la page des préférences Java et JSP.

CRWSA1295W Un kit JDK est manquant dans ce projet. Le kit JDK par défaut a été appliqué à ce projet.

Le kit JDK spécifié pour un projet importé n'a pas été reconnu par AppScan Source. Pour corriger cette erreur, ajoutez-le en tant que kit JDK reconnu à l'aide de la page des préférences Java et JSP.

CRWSA1300E Un kit JDK est manquant dans un ou plusieurs projets et aucun kit par défaut n'a été spécifié. Vous serez invité à choisir un kit JDK par défaut après avoir cliqué sur le bouton OK. A l'avenir, ce kit JDK sera utilisé pour tous les projets n'en ayant pas défini explicitement un.

CRWSA1305E L'application est en lecture seule. Les modifications apportées à l'évaluation ne persisteront pas dans les examens futurs.

Il est possible que l'application soit distante ou qu'elle nécessite une sauvegarde. Dans les deux cas, les modifications apportées à l'évaluation ne seront pas disponibles dans les examens futurs.

CRWSA1340E Echec de l'enregistrement de "{0}"

CRWSA1345E Le fichier d'application n'est pas accessible en écriture.

CRWSA1350E Erreur lors de la suppression d'entrées

CRWSA1355E Impossible de supprimer une application alors qu'une évaluation est en cours.

CRWSA1360E Impossible de supprimer un projet alors qu'une évaluation est en cours.

CRWSA1365W Tous les projets auxquels vous désirez appliquer les modifications devront être actualisés.

CRWSA1370W Certaines des constatations sélectionnées comportent déjà des notes. Etes-vous certain de vouloir les écraser ?

CRWSA1375W Echec de l'exportation vers le fichier

CRWSA1380W Il n'existe aucune constatation à exporter. Toutes les constatations sélectionnées sont absentes des résultats de l'évaluation.

CRWSA1383E Il n'existe aucune constatation à soumettre. Toutes les constatations sélectionnées ont déjà été soumises et la préférence "Soumettre les défauts une seule fois" a été activée.

CRWSA1385W Ces zones doivent être renseignées pour soumettre un défaut :

CRWSA1395W La sauvegarde du groupement vers {0} a abouti.

CRWSA1400E Le fichier de projet "{0}" est introuvable.

Vérifiez que le fichier de projet AppScan Source existe.

CRWSA1405E Le répertoire ASP.NET "{0}" est introuvable

Vérifiez que le répertoire ASP.NET existe.

CRWSA1410E Erreur lors de l'importation du fichier de projet "{0}"

CRWSA1415E Erreur lors de l'importation du répertoire ASP.NET "{0}"

CRWSA1420E Erreur lors de la création du projet

CRWSA1425E Le répertoire de travail "{0}" n'existe pas. Sélectionnez un répertoire valide.

CRWSA1430E La racine de contexte Web "{0}" n'existe pas. Sélectionnez un répertoire ou un fichier WAR valide.

CRWSA1435E La racine de contexte Web "{0}" ne peut pas être un répertoire parent ou le même répertoire que le répertoire de projet "{1}".

CRWSA1440E La racine de contenu "{0}" n'existe pas. Sélectionnez un répertoire valide.

CRWSA1445E Le fichier ou le répertoire "{0}" n'existe pas. Sélectionnez un fichier ou un répertoire valide.

CRWSA1450E La racine de contenu "{0}" ne peut pas être un répertoire parent ou le même répertoire que le répertoire de projet "{1}".

CRWSA1455E La méthode {0} comporte déjà un élément ActionObject pour le type {1}.

CRWSA1460E "{0}" n'est pas un répertoire valide

CRWSA1465E Kit JDK manquant pour le projet

CRWSA1470E Ce projet Java/JSP fait référence à un kit JDK ({0}) non valide. Un kit JDK par défaut sera fourni.

CRWSA1475E Le nom du kit JDK existe déjà

CRWSA1480E Il existe déjà un kit JDK nommé "{0}".

Spécifiez un nom de kit JDK unique.

CRWSA1485E Le nom du kit JDK doit comporter au moins un caractère.

CRWSA1490E Un kit JDK doit être sélectionné pour cette opération

Un kit JDK existant doit être sélectionné.

CRWSA1495E Sauvegarde impossible

CRWSA1500E Impossible de sauvegarder alors qu'une évaluation est en cours.

CRWSA1505E La racine source "{0}" existe déjà en tant que racine.

CRWSA1510E Le répertoire "{0}" n'existe pas. Sélectionnez un répertoire valide.

CRWSA1515E Le répertoire de précompilation "{0}" n'existe pas. Sélectionnez un répertoire valide.

CRWSA1520E La classe {0} existe à la fois dans la dépendance {1} et dans le chemin source.

Vérifiez les dépendances spécifiées.

CRWSA1525E Type de fichier inattendu dans le chemin de classes ({0}).

Vérifiez que votre projet Java contient uniquement des fichiers `.jar`, `.class` ou `.java`.

CRWSA1530E Les projets ne peuvent pas être validés alors qu'un examen est en cours.

CRWSA1535E

CRWSA1540E Le nom de configuration est vide

CRWSA1545E Racine source non valide

CRWSA1550E La racine source "{0}" est non valide. - Les racines source qui sont des fichiers doivent être situées sous le répertoire de travail.

CRWSA1555E Impossible de trouver une racine de contexte Web valide.

CRWSA1565E La création de la routine de codage pour {0} a abouti

CRWSA1568E Impossible de résoudre les variables dans le chemin "{0}"

Ajoutez une variable de chemin à l'aide de la page de préférence Modifier les variables.

CRWSA1570E Nom d'attribut non valide {0}

CRWSA1575E Nom d'attribut non valide {0}

CRWSA1580E Vous devez créer un attribut avant d'ajouter des valeurs.

CRWSA1585E Le nom de l'application ne doit pas être vide.

CRWSA1590E Entrée en double "{0}". Spécifiez une valeur unique.

CRWSA1595E Le nom ne peut pas être vide.

CRWSA1600E Une vue contenant des constatations doit être sélectionnée pour pouvoir générer un rapport.

CRWSA1625E Accès refusé. Le compte en cours ne dispose pas d'autorisation pour {0}.

CRWSA1630E Une erreur s'est produite lors de la tentative d'importation de l'espace de travail. Vérifiez que votre importateur est correctement configuré dans Préférences > Importateurs d'espace de travail Eclipse.

CRWSA1640E Impossible d'ajouter le projet. {0} est une application de référence ou en lecture seule.

L'application est en lecture seule ou il s'agit d'une référence à un espace de travail Eclipse ou à une solution Microsoft Visual Studio. Il est impossible d'ajouter les projets.

CRWSA1650E Impossible de trouver la feuille de style {0}.

CRWSA1653E Erreur

Cette erreur est suivie des deux points (:) et d'un message décrivant l'erreur. Ces documents fournissent une résolution possible pour certains de ces messages :

- Attempts to generate a report using the AppScan Source Maven integration result in error "Incorrect syntax for report option."
- Running ounceauto scanapplication results in "CRWSA1653E: You must log in first"
- Scanning results in "initializing manager applicationmanager" in AppScan Source for Security
- Publishing assessment to AppScan Enterprise using AppScan Source CLI fails with error code CRWSA1653E
- Publishing assessment from CLI results in CRWSA1653E Error

CRWSA1655E Un nom est requis pour le schéma

CRWSA1660E Au moins un critère doit être spécifié pour le schéma

CRWSA1665E Le critère ne peut pas être vide.

CRWSA1670E Un fichier est requis pour le schéma.

CRWSA1675E Il existe déjà une règle d'examen nommée "{0}". Choisissez un nom unique.

CRWSA1700E La présentation du rapport est actuellement vide. Ajoutez des éléments à l'onglet Agencement du rapport.

CRWSA1705E Il n'existe aucun groupement à exporter

CRWSA1710E Il n'existe aucune évaluation ouverte à exporter

CRWSA1715E Les données saisies dans les zones Nouveau mot de passe et Confirmation du nouveau mot de passe ne concordent pas.

CRWSA1725E Le mot de passe est trop court. Entrez un mot de passe comportant de 6 à 16 caractères.

CRWSA1730E Le mot de passe est trop long. Entrez un mot de passe comportant de 6 à 16 caractères.

CRWSA1735E Le fichier de paramètres "{0}" ne possède pas de nom d'affichage et sera ignoré.

CRWSA1736E Impossible d'ouvrir le fichier <nom_fichier>. Vérifiez que l'utilisateur exécutant le serveur d'automatisation possède les droits sur le fichier.

CRWSA1737E Impossible d'ouvrir le fichier <nom_fichier>. Vérifiez que l'utilisateur exécutant le serveur d'automatisation possède les droits sur le fichier et qu'il s'agit d'une application locale.

CRWSA1740E Le fichier de paramètres nommé "{0}" est déjà ouvert.

CRWSA1745E Le fichier de paramètres "{0}" n'est pas visible pour l'utilisateur.

CRWSA1750I Cet onglet contient certains paramètres qui requièrent un redémarrage des services AppScan Source pour que les modifications prennent effet. Avant de redémarrer les services AppScan Source, fermez toutes les applications client AppScan Source en cours d'exécution ou connectées à cet ordinateur. Souhaitez-vous redémarrer les services maintenant ?

CRWSA1755E "{0} comporte un paramètre pour lequel un ou plusieurs attributs obligatoires sont manquants (nom, nom d'affichage ou type). Ce fichier ne sera pas chargé."

CRWSA1760E Une erreur de licence s'est produite :

CRWSA1775E L'initialisation de la connexion à la console AppScan Enterprise Console a échoué. Vérifiez la disponibilité de la console AppScan Enterprise Console avec ces paramètres dans le navigateur externe.

CRWSA1780E L'initialisation de la connexion à la console AppScan Enterprise Console a échoué. Consultez les journaux et/ou vérifiez la disponibilité de la console AppScan Enterprise Console avec ces paramètres dans le navigateur externe. - Détail de l'erreur : - {0}

CRWSA1790E Echec de la publication de l'évaluation sur la console AppScan Enterprise Console. Consultez les journaux et/ou contactez votre administrateur système.

CRWSA1795E Echec de l'initialisation du service de publication. Vérifiez vos paramètres de connexion à la console AppScan Enterprise Console.

CRWSA1800E Echec de l'initialisation du service de publication. Vérifiez la disponibilité de la console AppScan Enterprise console et votre capacité à vous connecter.

CRWSA1805E Impossible de télécharger le fichier. - Une erreur s'est produite lors de la tentative de connexion au site.

Vérifiez que vous disposez d'une connexion Internet fonctionnelle et que le site est disponible.

CRWSA1810E Vous devez vous connecter pour pouvoir ouvrir le groupement créé dans la version 7.0.0 du produit

CRWSA9999E AppScan Source for Analysis a rencontré une erreur critique et va être fermé.

Chapitre 9. Installation et emplacements des fichiers de données utilisateur

Lorsque vous installez AppScan Source, les fichiers de données utilisateur et de configuration sont stockés hors du répertoire d'installation.

- «Répertoire d'installation par défaut»
- «Répertoire de données AppScan Source par défaut»
- «Emplacement des fichiers temporaires AppScan Source», à la page 152

Répertoire d'installation par défaut

Lorsque AppScan Source est installé, le logiciel est placé dans l'un des emplacements par défaut suivants :

- Versions 32 bits de Microsoft Windows :
  `<UNITESYSTEME>:\Program Files\IBM\AppScanSource`
- Versions 64 bits de Microsoft Windows :
  `<UNITESYSTEME>:\Program Files (x86)\IBM\AppScanSource`
- Linux : si vous êtes l'utilisateur root, l'assistant d'installation installe votre logiciel dans `/opt/ibm/appscansource`. Si vous n'êtes pas l'utilisateur root, vous pouvez installer le plug-in Eclipse AppScan Source for Development, qui s'installe dans `<répertoire_accueil>/AppScan_Source` par défaut.
- macOS : `/Applications/AppScanSource.app`

Important :

- Le nom du répertoire d'installation peut uniquement contenir des caractères anglais. Les noms de dossiers contenant des caractères autres qu'anglais ne sont pas autorisés.
- Si vous effectuez une installation sur Windows, vous devez disposer des privilèges d'administrateur pour installer les composants AppScan Source.
- Si vous effectuez une installation sur Linux, vous devez disposer des privilèges root pour installer les composants serveur de AppScan Source.

Répertoire de données AppScan Source par défaut

Les données AppScan Source sont des fichiers de configuration, exemples et journaux. Lorsque AppScan Source est installé, les fichiers de données sont placés dans ces emplacements par défaut :

- Microsoft Windows : `<UNITESYSTEME>:\ProgramData\IBM\AppScanSource`

  Remarque : `ProgramData\` est un dossier masqué et, pour l'afficher, vous devez modifier les préférences d'affichage dans l'**explorateur** afin d'afficher les fichiers et les dossiers masqués.
- Linux : `/var/opt/ibm/appscansource`
- macOS : `/Users/Shared/AppScanSource`

Pour savoir comment modifier l'emplacement du répertoire de données AppScan Source, voir «Modification du répertoire de données AppScan Source», à la page 152.

Emplacement des fichiers temporaires AppScan Source

Certaines opérations AppScan Source génèrent la création de fichiers temporaires qui sont stockés dans les emplacements suivants par défaut :

- Microsoft Windows: `<UNITESYSTEME>:\ProgramData\IBM\AppScanSource\temp`

  Remarque : `ProgramData\` est un dossier masqué et, pour l'afficher, vous devez modifier les préférences d'affichage dans l'**explorateur** afin d'afficher les fichiers et les dossiers masqués.
- Linux : `/var/opt/ibm/appscansource/temp`
- macOS : `/Users/Shared/AppScanSource/temp`

L'emplacement des fichiers temporaires est toujours dans un répertoire `temp`, dans le répertoire de données AppScan Source. Vous pouvez changer l'emplacement des fichiers temporaires en modifiant le répertoire de données, comme décrit dans «Modification du répertoire de données AppScan Source». Le répertoire `temp` est alors situé dans le répertoire de données que vous avez sélectionné.

Modification du répertoire de données AppScan Source

Vous pouvez modifier l'emplacement du répertoire de données AppScan Source à des fins de gestion de l'espace du disque dur. Vous pouvez modifier l'emplacement après l'installation de AppScan Source en appliquant les étapes de cette rubrique.

Avant de commencer

Avant d'effectuer cette tâche, vérifiez que toutes les applications client AppScan Source ont été fermées ou arrêtées. Les applications client AppScan Source sont les suivantes :

- AppScan Source for Analysis
- AppScan Source for Development (plug-in Eclipse ou Visual Studio) (pris en charge uniquement sous Windows et Linux)
- interface de ligne de commande (CLI) d'AppScan Source
- AppScan Source for Automation

En outre, si vous avez installé AppScan Source for Automation, vérifiez que l'Automation Server a été arrêté :

- Sous Windows, arrêtez le service **IBM Security AppScan Source Automation**.
- Sous Linux, lancez la commande suivante : `/etc/init.d/ounceautod stop`
- Sous macOS, exécutez cette commande : `launchctl stop com.ibm.appscan.autod`

Procédure

1. Définissez une variable d'environnement `APPSCAN_SOURCE_SHARED_DATA=<data_dir>`, où `<data_dir>` est l'emplacement dans lequel vous souhaitez stocker les données AppScan Source.

   Remarque :
   - L'emplacement `<data_dir>` doit être un chemin d'accès absolu et complet qui existe sur la même machine que votre installation AppScan Source.
   - Le nom du répertoire `<data_dir>` peut uniquement contenir des caractères anglais. Les dossiers portant des noms contenant des caractères non anglais ne sont pas autorisés.

2. Localisez le répertoire de données par défaut ayant été créé lorsque AppScan Source a été installé (voir «Répertoire de données AppScan Source par défaut», à la page 151 pour plus de détails sur les emplacements de répertoire de données par défaut).
3. Copiez ou déplacez le contenu du répertoire de données par défaut vers l'emplacement <data_dir> spécifié dans la variable d'environnement.
4. **Applicable uniquement à AppScan Source for Automation installé sous Linux** :
   a. Editez le fichier /etc/init.d/ounceautod.
   b. Localisez la ligne suivante

      ```
      su - ounce -c
      'export LD_LIBRARY_PATH="/opt/IBM/AppScan_Source/bin":$LD_LIBRARY_PATH &&
      cd "/opt/IBM/AppScan_Source/bin" &&
      "/opt/IBM/AppScan_Source/bin/ounceautod" -s' >>
      "/var/opt/ibm/appscansource/logs/ounceautod_output.log" 2>&1 &
      ```

      et remplacez-la par :

      ```
      su - ounce -c
      'export APPSCAN_SOURCE_SHARED_DATA=<new data directory path here> &&
      export LD_LIBRARY_PATH="/opt/IBM/AppScan_Source/bin":$LD_LIBRARY_PATH &&
      cd "/opt/IBM/AppScan_Source/bin" &&
      "/opt/IBM/AppScan_Source/bin/ounceautod" -s' >>
      "<new data directory path here>/logs/ounceautod_output.log" 2>&1 &
      ```

      Remarque : La commande ci-dessus est sur une ligne.
   c. Sauvegardez le fichier /etc/init.d/ounceautod.

Que faire ensuite

Si vous avez installé AppScan Source for Automation, démarrez le Automation Server :

- Sous Windows, démarrez le service **IBM Security AppScan Source Automation**.
- Sous Linux, lancez la commande suivante : /etc/init.d/ounceautod start
- Sous macOS, exécutez cette commande : launchctl start com.ibm.appscan.autod

Classifications

Les constatations sont classées par AppScan Source pour indiquer s'il s'agit de constatations de sécurité ou de couverture d'examen. Les constatations de sécurité représentent des vulnérabilités réelles ou probables en matière de sécurité, alors que les constatations de couverture d'examen représentent les zones dans lesquelles la configuration pourrait être améliorée afin de fournir une meilleure couverture d'examen.

Chaque constatation (ou résultat) se rattache à l'une de ces *classifications* :

- Constatation de sécurité **définitive** : Une constatation contenant une conception définitive, une implémentation ou une violation de stratégie qui présente une possibilité pour un agresseur de provoquer un fonctionnement imprévu de l'application.

  Cette attaque peut résulter en un accès non autorisé, le vol ou l'altération de données, de systèmes ou de ressources. Chaque constatation de sécurité définitive est complètement décomposée et le schéma sous-jacent spécifique de la condition vulnérable est connu et décrit.

- Constatation de sécurité **suspectée** : Une constatation qui indique une condition suspecte et potentiellement vulnérable requérant des informations ou des investigations supplémentaires. Un élément ou une structure de code qui peut créer une vulnérabilité en cas d'utilisation incorrecte.

  Une constatation suspectée diffère d'une constatation définitive car il existe une condition inconnue qui empêche la détermination définitive d'une vulnérabilité. L'utilisation d'éléments dynamiques ou de fonctions de bibliothèque pour lesquelles aucun code source n'est disponible sont des exemples de cette incertitude. En conséquence, un niveau de recherche supplémentaire s'impose pour confirmer ou rejeter une constatation suspectée comme étant définitive.
- Constatation de **couverture d'examen** : Constatations qui représentent les zones dans lesquelles la configuration pourrait être améliorée afin de fournir une meilleure couverture d'examen (par exemple, les constatations de collecteur indéterminé).

Remarque : Dans certains cas, la classification **Aucun** est utilisée pour indiquer une constatation qui n'est ni une constatation de sécurité ni une constatation de couverture d'examen.

Remarques

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Pour le Canada, veuillez adresser votre courrier à :

IBM Director of Commercial Relations
IBM Canada Ltd.
3600 Steeles Avenue East
Markham, Ontario
L3R 9Z7 Canada

Les informations sur les licences concernant les produits utilisant un jeu de caractères double octet peuvent être obtenues par écrit à l'adresse suivante :

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales.

LE PRESENT DOCUMENT EST LIVRE EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE.

Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut également, à tout moment et sans préavis, apporter des améliorations et/ou des modifications aux produits et/ou programmes décrits sur le site.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758
U.S.A.

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans ce document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM conformément aux dispositions de l'ICA, des Conditions internationales d'utilisation des logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Tous les tarifs indiqués sont les prix de vente actuels suggérés par IBM et sont susceptibles d'être modifiés sans préavis. Les tarifs appliqués peuvent varier selon les revendeurs.

Ces informations sont fournies uniquement à titre de planification. Elles sont susceptibles d'être modifiées avant la mise à disposition des produits décrits.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

LICENCE DE COPYRIGHT :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation IBM.

Toute copie totale ou partielle de ces programmes exemples et des oeuvres qui en sont dérivées doit comprendre une notice de copyright, libellée comme suit :

© (nom de votre société) (année). Des segments de code sont dérivés des Programmes exemples d'IBM Corp. © Copyright IBM Corp. _indiquez l'année ou les années_. All rights reserved.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

Marques

IBM, le logo IBM et ibm.com sont des marques d'International Business Machines Corp. dans de nombreux pays. Les autres noms de produits et de services peuvent être des marques d'IBM ou d'autres sociétés. La liste actualisée de toutes les marques d'IBM est disponible sur la page Web Copyright and trademark information à l'adresse www.ibm.com/legal/copytrade.shtml.

Adobe, Acrobat, PostScript et toutes les marques incluant Adobe sont des marques d'Adobe Systems Incorporated aux Etats-Unis et/ou dans certains autres pays.

IT Infrastructure Library est une marque de The Central Computer and Telecommunications Agency qui fait désormais partie de The Office of Government Commerce.

Intel, le logo Intel, Intel Inside, le logo Intel Inside, Intel Centrino, le logo Intel Centrino, Celeron, Intel Xeon, Intel SpeedStep, Itanium, et Pentium sont des marques d'Intel Corporation ou de ses filiales aux Etats-Unis et dans certains autres pays.

Linux est une marque de Linus Torvalds aux Etats-Unis et/ou dans certains autres pays.

Microsoft, Windows, Windows NT et le logo Windows sont des marques de Microsoft Corporation aux Etats-Unis et/ou dans certains autres pays.

ITIL est une marque de The Office of Government Commerce et est enregistrée au bureau américain Patent and Trademark Office.

UNIX est une marque enregistrée de The Open Group aux Etats-Unis et/ou dans certains autres pays.

Java ainsi que tous les logos et toutes les marques incluant Java sont des marques d'Oracle et/ou de ses filiales.

Cell Broadband Engine est une marque de Sony Computer Entertainment, Inc., aux Etats-Unis et/ou dans certains autres pays, et y est utilisé sous licence.

Linear Tape-Open, LTO, le logo LTO, Ultrium et le logo Ultrium sont des marques de HP, IBM Corp. et Quantum aux Etats-Unis et/ou dans certains autres pays.

Index

L

M

O

P

R

T

U

W