RICOH

RICOH Transforms Suite

Outil PDF2PDFUA Guide utilisateur

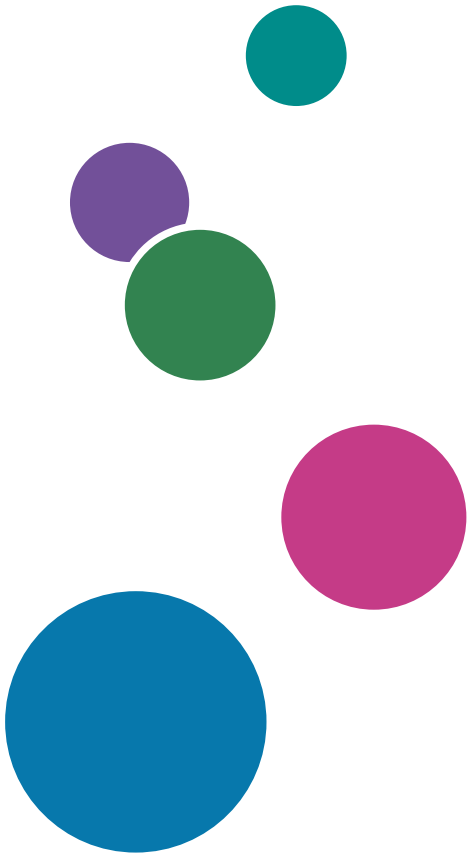
Version 1.5

Utilisation de l'outil PDF2PDFUA 1

Garantir la conformité aux normes PDF/UA et WCAG 2

Interface de programmation d'applications Java 3

Pour obtenir des informations non disponibles dans ce manuel, reportez-vous au système d'aide de votre produit.



SOMMAIRE

Introduction

Important ... 2
Précautions concernant ce guide ... 2
Publications relatives à ce produit ... 2
Symboles ... 2
Abréviations ... 3
Marques ... 3

1 Utilisation de l'outil PDF2PDFUA

Utilisation de l'outil PDF2PDFUA via la transformation AFP2PDF Plus ... 5
Utilisation de l'outil PDF2PDFUA depuis la ligne de commande ... 5
Syntaxe ... 5
Paramètres ... 6
Configuration du fichier d'options `PDF2PDFUA.cfg` ... 7
Fonctionnalités prises en charge par la transformation PDF2PDFUA ... 8

2 Garantir la conformité aux normes PDF/UA et WCAG

Exigences relatives aux fichiers d'entrée pour la conformité à la norme PDF/UA ... 9
Validation du fichier PDF d'entrée et du fichier PDF UA de sortie ... 9
Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.0 (niveau AA) ... 9
Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.0 (niveau AAA) ... 10
Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.1 (niveau AA) ... 10
Vérification de la conformité aux normes PDF/UA ou WCAG ... 10

3 Interface de programmation d'applications Java

Classes ... 11
Exemples ... 15

Introduction

Important

Dans la mesure maximale autorisée par les lois en vigueur, le fabricant ne pourra en aucun cas être tenu responsable de tout dommage, quel qu'il soit, résultant de défaillances de ce produit, de pertes de documents ou de données, ou de l'utilisation ou de la non-utilisation de ce produit et des manuels d'utilisation fournis avec celui-ci.

Assurez-vous de toujours copier les documents importants et les données ou d'en faire une sauvegarde. Les documents ou les données peuvent être effacés en raison de vos erreurs de manipulation ou de dysfonctionnements du logiciel. De plus, il est de votre responsabilité de prendre les mesures de protection appropriées contre les virus informatiques, les vers et autres logiciels malveillants.

En aucun cas le fabricant sera tenu responsable des documents que vous avez créés à l'aide de ce produit ou des résultats issus des données que vous avez exécutées.

Précautions concernant ce guide

- Certaines illustrations ou explications de ce guide peuvent différer de votre produit en raison d'une amélioration ou d'une modification du produit.
- Le contenu de ce document est susceptible d'être modifié sans préavis.
- Aucune partie de ce document ne peut être dupliquée, répliquée, reproduite sous quelque forme que ce soit, modifiée ou citée sans le consentement préalable du fournisseur.

Publications relatives à ce produit

Cette publication présente une vue d'ensemble de l'outil PDF2PDFUA et fournit des informations sur son installation et son utilisation.

Manuels d'instruction

Pour plus d'informations sur l'outil PDF2PDFUA, veuillez consulter les documents suivants :

- *PDF2PDFUA Manuel d'installation*
- *PDF2PDFUA Guide utilisateur*

Symboles

Les symboles suivants sont utilisés dans le présent manuel afin de vous aider à identifier le contenu rapidement.



Ce symbole indique les points auxquels vous devez apporter une attention particulière lors de l'utilisation de l'application. Lisez consciencieusement ces explications.

Remarque

Ce symbole indique des informations supplémentaires susceptibles de vous être utiles, mais qui ne sont pas indispensables à la réalisation d'une tâche.

[Gras]

L'utilisation du **[Gras]** à l'intérieur des crochets sert à désigner les noms de menus, éléments de menu, paramètres, libellés de champs, boutons et touches.

`Italique`

Les *`caractères en italique`* désignent les variables que vous devez remplacer par vos propres informations.

`Chasse fixe`

Les `caractères à chasse fixe` sont utilisés pour les entrées et sorties informatiques.

...

Les points de suspension correspondent à une série qui peut continuer.

Abréviations

AES

Advanced Encryption Standard

AFP

Advanced Function Presentation

PDF

Portable Document Format

UA

Accessibilité universelle

WCAG

Directives relatives à l'accessibilité des contenus Web

Marques

Adobe, le logo Adobe, PostScript et le logo PostScript, PDF et le logo PDF sont des marques déposées ou des marques commerciales d'Adobe aux États-Unis et/ou dans d'autres pays.

Linux est la marque déposée de Linus Torvalds aux États-Unis et/ou d'autres pays.

Microsoft, Windows, Windows NT et le logo Windows sont des marques commerciales de Microsoft Corporation aux États-Unis et/ou dans d'autres pays.

Sentinel est une marque déposée de Thales DIS CPL USA, Inc.

UNIX est une marque de The Open Group aux États-Unis et dans certains autres pays.

Les autres noms de sociétés, de produits et de services peuvent appartenir à des tiers.

1. Utilisation de l'outil PDF2PDFUA

- **Utilisation de l'outil PDF2PDFUA via la transformation AFP2PDF Plus**
- **Utilisation de l'outil PDF2PDFUA depuis la ligne de commande**
- **Syntaxe**
- **Paramètres**
- **Configuration du fichier d'options** `PDF2PDFUA.cfg`
- **Fonctionnalités prises en charge par la transformation PDF2PDFUA**

Utilisation de l'outil PDF2PDFUA via la transformation AFP2PDF Plus

Pour convertir des fichiers AFP en documents PDF/UA, vous devez ajouter ce paramètre d'option dans le fichier de configuration d'AFP2PDF Plus :

PDFUA_CONTROL_FILE=*`fichier`*

Indique le nom complet du fichier de contrôle de l'outil PDF2PDFUA. Ce fichier est un fichier XML qui contient des informations sur le contenu du fichier d'entrée. L'outil PDF2PDFUA utilise le document pour baliser son contenu conformément aux exigences de la norme PDF/UA.

PDFUA_CACHE_SIZE=*`nombre`*

Indique le nombre de fichiers de contrôle PDFUA les plus utilisés stockés dans le cache. Ce nombre doit être supérieur à 0.

Remarque

- Pour obtenir un fichier de sortie PDF/UA chiffré ou linéarisé, configurez les paramètres du fichier d'options AFP2PDF Plus lors de l'exécution de la transformation AFP2PDF Plus :

  Pour le chiffrement :

  `Default_Encryption_Permissions`

  `Default_owner_password`

  `Default_user_passowrd`

  Pour la création de documents linéarisés :

  `Default_linearization`

  Pour plus d'informations sur la configuration de ces options, veuillez consulter le chapitre *`Fichier d'options de transformation AFP2PDF Plus`* d'*`AFP2PDF Plus Guide utilisateur`*.

Limitation :

- Lors de la création de fichiers PDF/UA, le chiffrement AES 256 bits n'est pas pris en charge.

Utilisation de l'outil PDF2PDFUA depuis la ligne de commande

Lorsque vous exécutez l'outil PDF2PDFUA à partir de la ligne de commande, le fichier d'entrée peut être soit un document PDF, soit un fichier de contrôle converti par l'outil PDF2PDFUA en un fichier conforme à la norme PDF/UA.

Syntaxe

Selon le système d'exploitation, la syntaxe de la transformation PDF2PDFUA est la suivante :

pdf2pdfua.*nnn* [**-o** *nom du fichier de sortie*] [**-i** [*nom du fichier de contrôle*] [**-b** [*nombre_de_threads*] *répertoire d'entrée répertoire de sortie*] [**-d** *sortie pour l'extraction du texte*] [**-linearize** *true/false*] [**-chiffrer** *mot de passe de propriétaire mot de passe utilisateur* indicateurs de chiffrement] [**-decrypt** *mot de passe*] [**-key** *mot de passe*] [**-trace** *expression régulière*] [*fichier d'entrée*]

1

où *.nnn* correspond à l'extension **.bat** pour Windows et à l'extension **.sh** pour Linux et AIX.

Paramètres

-b

Ce paramètre sert à exécuter la transformation PDF2PDFUA en mode lot. La transformation convertit tous les fichiers PDF d'un répertoire en fichiers PDF/UA à l'aide d'un fichier de contrôle unique, spécifié via le paramètre **-i**.

Pour spécifier le nombre de threads, indiquez ce nombre immédiatement après l'option **-b**. Le nombre de threads par défaut est 1.

Remarque

- En mode lot, les paramètres *répertoire d'entrée* et *répertoire de sortie* sont obligatoires. Le paramètre *répertoire d'entrée* remplace le paramètre *fichier d'entrée*.

-linearize

Ce paramètre est facultatif.

La valeur par défaut n'est pas précisée.

- Si la valeur est « true », la sortie est linéarisée.
- Si la valeur est « false », la sortie n'est pas linéarisée, même si l'entrée l'est.
- Si aucune valeur n'est spécifiée, la sortie n'est linéarisée que si l'entrée est linéarisée.

-encrypt

Ce paramètre est facultatif.

Par défaut, l'utilisateur ne dispose d'aucune autorisation. Les indicateurs suivants sont utilisés pour accorder des autorisations :

`-allowAnnotations`

`-allowAssemble`

`-allowCopy`

`-allowFill`

`-allowModify`

`-allowPrintDegraded`

`-allowPrintHigh`

`-allowPrint`, `-allowPrintDegraded` ou `-allowPrintHigh`.

-decrypt *mot de passe*

Ce paramètre supprime le chiffrement d'origine.

-key *mot de passe*

Ce paramètre permet d'ouvrir les documents verrouillés sans supprimer le chiffrement d'origine. Les droits d'accès sont conservés.

-trace *expression régulière*

Ce paramètre sert au débogage des fichiers de contrôle. Tous les éléments de traçage sont activés par défaut en mode ligne de commande.

-validate

Ce paramètre active le validateur. Il est possible de l'utiliser sans fichier de contrôle si vous effectuez une pré-validation du fichier PDF d'entrée. Si un fichier de contrôle est spécifié, le système vérifie également le fichier PDF généré.

Configuration du fichier d'options PDF2PDFUA.cfg

Les paramètres supplémentaires permettant de contrôler les réglages de la transformation PDF2PDFUA sont définis dans un fichier d'options. Ce fichier s'appelle PDF2PDFUA.cfg. Le fichier PDF2PDFUA.cfg doit se trouver dans le répertoire racine de l'outil PDF2PDFUA.

Les paramètres de l'option PDF2PDFUA.cfg sont les suivants :

AlwaysWriteProducer=TRUE | FALSE

Indique que le fichier PDF de sortie doit contenir des informations concernant la version de la transformation PDF2PDFUA et la version du fichier de contrôle dans le champ Éditeur de fichiers PDF.

La valeur par défaut est **False**.

Lorsque la valeur est **False**, le paramètre remplit la même fonction que celle spécifiée dans le fichier de contrôle.

Lorsque la valeur est **True**, le paramètre inclut les métadonnées du producteur, quelles que soient les spécifications du fichier de contrôle. Les métadonnées sont utiles pour le développement. Ce paramètre n'est plus nécessaire, car les versions récentes de PDF2PDFUA intègrent les métadonnées selon des méthodes non standard. Les développeurs peuvent extraire les métadonnées, mais un lecteur de PDF ne tiendra tout simplement pas compte de celles-ci.

DefaultLogSize= integer

La valeur par défaut est **10**. Ce paramètre correspond à la taille maximale du fichier journal par défaut, en mégaoctets. Ce paramètre n'a aucun effet si la valeur de **useDefaultLog** est « false ».

errProgramVersion = true|false

Définit la manière dont la transformation traite les fichiers de contrôle conçus pour les versions plus récentes de PDF2PDFUA.

Lorsque cette valeur est **true**, une exception est levée, ce qui empêche la transformation d'exécuter des fichiers de contrôle incompatibles.

Lorsque cette valeur est définie sur **false**, le programme tente d'exécuter des fichiers de contrôle incompatibles, ce qui génère un avertissement dans le gestionnaire d'impression (STDOUT par défaut) et dans les journaux. Cela pourrait entraîner la génération de fichiers PDF incorrects.

La valeur par défaut est **true** et il est vivement recommandé de conserver cette option sur **true**.

LegacyLinePosition=TRUE | FALSE

La valeur par défaut est **False**.

Lorsque la valeur est **False**, la position est déterminée par le mot dont la coordonnée Y est la plus petite.

Lorsque la valeur est **True**, la position est déterminée par le mot dont la coordonnée Y est la plus élevée. Si la ligne contient des indices, l'opérateur obtenirPositionDeLigne renvoie la coordonnée Y de l'indice.

LegacyNextLine= TRUE | FALSE

La valeur par défaut est **True**. Cela affecte le comportement des méthodes **obtenirLaLigneSuivante** et **obtenirLaLigneSuivanteDansLaRégion**, à l'exception des méthodes ligneSuivante et ligneSuivanteDansLaRégion. Ces opérateurs utilisent toujours le nouveau comportement.

Dans le cadre du comportement historique, le programme recherche la première ligne pour laquelle y(ligne) > y(curseur). Cela ne fonctionne pas correctement si la position du curseur est définie à l'aide d'**obtenirPositionDeLigne**.

Dans le cadre de ce nouveau comportement : il recherche la première ligne pour laquelle y(ligne) ≥ y(curseur). Cela corrige certains problèmes, mais en introduit un autre :

```
if y(line) = y(cursor)
```

Par exemple :

```
<Cursor> <Y> <Expr> getLinePosition(getNextLine) </Expr> </Y> </Cursor>
<Expr> $line1 = getNextLine </Expr>
<Expr> $line2 = getCurrentLine </Expr>
```

Ici, ligne 1 et ligne 2 sont identiques.

UseDefaultLog= TRUE | FALSE

La valeur par défaut est **True**.

Lorsque la valeur est **True**, cela crée un fichier journal par défaut si aucun n'a été transmis par l'API. Il est utile en mode autonome.

Lorsque la valeur est **False**, la journalisation n'est activée que si elle a été configurée via l'API.

Fonctionnalités prises en charge par la transformation PDF2PDFUA

Grâce à la transformation PDF2PDFUA Plus, vous pouvez convertir par lots plusieurs fichiers d'entrée partageant un même modèle de génération. Vous pouvez créer un fichier de contrôle pour n'importe quel fichier PDF, qu'il s'agisse de factures, de relevés de compte ou de relevés de carte bancaire. Vous avez la possibilité de créer un fichier de contrôle pour chaque type de facture.

PDF2PDFUA Plus offre une prise en charge améliorée de la gestion des données variables. PDF2PDFUA Plus est capable de traiter des données variables situées à des emplacements fixes, telles que les noms ou les champs d'adresse, mais aussi des tables comportant un nombre variable de lignes ou s'étendant sur plusieurs pages. La transformation PDF2PDFUA vous permet de ne marquer que le contenu réel et d'ignorer les artefacts non pertinents.

Remarque

Veuillez contacter votre représentant Ricoh pour obtenir de plus amples informations concernant la procédure de création d'un fichier de contrôle.

2. Garantir la conformité aux normes PDF/UA et WCAG

- **Exigences relatives aux fichiers d'entrée pour la conformité à la norme PDF/UA**
- **Validation du fichier PDF d'entrée et du fichier PDF UA de sortie**
- **Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.0 (niveau AA)**
- **Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.0 (niveau AAA)**
- **Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.1 (niveau AA)**
- **Vérification de la conformité aux normes PDF/UA ou WCAG**

Exigences relatives aux fichiers d'entrée pour la conformité à la norme PDF/UA

Si le fichier d'entrée présente l'un de ces problèmes, il ne pourra pas être traité ou le fichier résultant ne sera pas conforme à la norme PDF/UA.

- Le fichier PDF contient du contenu qui scintille ou des éléments PDF à l'origine d'un scintillement (Protocole Matterhorn – Ch03-X).
- Les informations contenues dans le fichier PDF sont transmises par le biais du contraste, de la couleur ou de la mise en page (MP Ch04).
- Le fichier PDF contient un fichier audio (MP Ch05).
- Le fichier PDF utilise des codes de caractères sans correspondance Unicode (MP Ch10).
- Le fichier PDF contient des images accompagnées de paragraphes de texte (MP C13-06).
- Le fichier PDF utilise des polices non intégrées (MP C21).

  Remarque

  Si vous l'utilisez avec la transformation AFP2PDF Plus, veillez à ce que celle-ci soit configurée pour intégrer les polices.
- Le fichier PDF contient des formulaires.
- Le fichier PDF contient des éléments interactifs.

Validation du fichier PDF d'entrée et du fichier PDF UA de sortie

Le validateur PDF2PDFUA génère des avertissements pouvant faire référence à des index spécifiques issus du protocole Matterhorn 1.1 des associations PDF. Le pré-validateur vérifie certaines des conditions mentionnées dans le protocole Matterhorn.

Pour obtenir plus d'informations sur les erreurs affichées, recherchez l'identifiant de l'erreur dans le document de protocole Marterhorn.

Veuillez contacter votre éditeur de fichiers PDF afin d'analyser et de résoudre cette erreur PDF et d'obtenir un fichier de sortie compatible avec la norme PDF/UA.

Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.0 (niveau AA)

Pour vous assurer que le fichier d'entrée est conforme aux normes d'accessibilité définies dans les Directives pour l'accessibilité des contenus Web (WCAG) 2.0, niveau AA, veuillez vérifier les points suivants :

WCAG 1.4.3 : Contraste minimal

Le fichier d'entrée doit contenir du texte et des images présentant un contraste minimal de 4,5:1.

WCAG 1.4.5 : Images en tant que texte

Le fichier d'entrée doit fournir des informations sous forme de texte, et non sous forme d'images contenant du texte.

Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.0 (niveau AAA)

Pour vous assurer que le fichier d'entrée est conforme aux normes d'accessibilité définies dans les Directives pour l'accessibilité des contenus Web (WCAG) 2.0, niveau AA, veuillez vérifier les points suivants :

WCAG 1.4.6 – Exigence relative au contraste renforcé

La présentation visuelle du texte et des images contenant du texte présente un rapport de contraste d'au moins 7:1.

WCAG 1.4.8 – Présentation visuelle

Le fichier d'entrée doit répondre aux exigences suivantes :

1. La largeur ne dépasse pas 80 caractères ou glyphes (40 pour les caractères CJK).
2. Le texte n'est pas justifié (aligné à la fois sur la marge de gauche et sur la marge de droite).
3. L'interligne au sein des paragraphes est d'au moins un espace et demi, et l'espacement entre les paragraphes est au moins 1,5 fois supérieur à l'interligne.

WCAG 1.4.9 Images de texte (aucune exception)

Les images représentant du texte ne sont utilisées qu'à des fins purement décoratives ou lorsque la mise en forme particulière du texte est essentielle à la compréhension de l'information transmise.

Exigences relatives aux fichiers d'entrée pour la conformité aux WCAG 2.1 (niveau AA)

1.4.11 Contraste des éléments non textuels (niveau AA)

La présentation visuelle des objets graphiques présente un rapport de contraste d'au moins 3:1 par rapport aux couleurs adjacentes.

Vérification de la conformité aux normes PDF/UA ou WCAG

Vous pouvez utiliser ces outils pour vous assurer que votre fichier PDF est conforme aux normes PDF/UA ou WCAG :

PAC – Outil de vérification de l'accessibilité des fichiers PDF

Un outil téléchargeable gratuitement qui permet de vérifier si un fichier PDF est conforme aux normes PDF/UA ou aux WCAG. Si la vérification échoue, vous pouvez consulter les problèmes qui ont entraîné la non-conformité du fichier. Vous pouvez vous le procurer en ligne auprès de la Fondation PDF/UA : https://pac.pdf-accessibility.org/en.

3. Interface de programmation d'applications Java

- **Classes**

Ce chapitre décrit l'API Java fournie par RICOH PDF2PDFUA.

Le code entre crochets angulaires fait référence à un élément XML dans le fichier de contrôle. Par exemple, `<If>` fait référence aux conditions dans le fichier de contrôle.

Classes

Ces classes font partie de l'interface de programmation d'applications (API) Java de RICOH PDF2PDFUA.

enum EncryptionAlgorithm

Package: **com.ricoh.pdf2pdfua.api**

Option permettant de définir l'algorithme de chiffrement du fichier PDF généré.

Valeurs :

none

La sortie n'est pas chiffrée.

preserve

Conserve le chiffrement du fichier PDF d'origine.

Standard128

Un chiffrement par flux utilisant une clé de 128 bits selon l'algorithme RC4.

AES128

Norme de chiffrement avancée utilisant une clé de 128 bits.

AES256

Norme de chiffrement avancée utilisant une clé de 256 bits.

interface IControlFile

Package: **com.ricoh.pdf2pdfua.api**.

Représente l'arbre syntaxique du fichier de contrôle. Le fichier de contrôle est analysé une seule fois, puis appliqué à plusieurs fichiers PDF.

```
IControlFile refresh() throws PreprocessingException;
```

Permet de vérifier si le fichier de contrôle a subi des modifications.

Renvoie `this` si le fichier de contrôle est à jour.

Renvoie un nouveau fichier `IControlFile` si les fichiers sous-jacents ne sont plus synchronisés.

interface Pdfua

Package: **com.ricoh.pdf2pdfua.api**

L'interface SPI (Service Provider Interface) Java pour RICOH PDF2PDFUA.

```
IControlFile newControlFile(String path);
```

Analyse le fichier de contrôle indiqué par « path ». Une même instance de fichier de contrôle peut être réutilisée dans plusieurs threads.

```
PdfUaOpt newOpt();
```

Crée une instance de PdfUaOpt avec les paramètres par défaut.

```
void run(FileInputStream fis, File out, IControlFile cf, PdfUaOpt opt) throws PDFUAExcepti
```

Cette méthode lance la conversion PDF2PDFUA. Il est thread-safe et réentrant.

```
byte[] run(InputStream inPdf, IControlFile cf, PdfUaOpt opt) throws PDFUAException;
```


Cette méthode exécute la conversion PDF2PDFUA et renvoie une table d'octets contenant le fichier PDF balisé. Il est thread-safe et réentrant.

class PDFUAException extends Exception

Package: **com.ricoh.pdf2pdfua.api**

Classe de base pour les erreurs de conversion PDF2PDFUA.

class PdfUaLoader

Package: **com.ricoh.pdf2pdfua.api**

Petit utilitaire qui charge l'implémentation du service PDF2PDFUA. Le service peut être chargé directement à l'aide d'un ServiceLoader.

```
public static Pdfua load();
```

Charge l'implémentation du service PDF2PDFUA.

abstract class PdfUaOpt

Package: **com.ricoh.pdf2pdfua.api**

Permet de configurer le comportement de PDF2PDFUA. Il s'agit d'un conteneur de configuration qui ne comporte que des méthodes getter et setter.

Il s'agit d'une classe abstraite, mais elle ne doit en aucun cas être étendue par les utilisateurs de l'API. Pour créer une instance, utilisez `Pdfua.newOpt()`.

Une même instance peut être utilisée par plusieurs threads, mais elle ne doit pas être modifiée pendant l'exécution de PDF2PDFUA.

```
public boolean getAllowAnnotations();
public void setAllowAnnotations(boolean allowAnnotation);
```

Indicateur d'autorisation pour le chiffrement des fichiers PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : false

```
public boolean getAllowAssemble();
public void setAllowAssemble(boolean allowAssemble);
```

Indicateur d'autorisation pour le chiffrement des fichiers PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : false

```
public boolean getAllowCopy();
public void setAllowCopy(boolean allowCopy);
```

Indicateur d'autorisation pour le chiffrement des fichiers PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : false

```
public boolean getAllowFill();
public void setAllowFill(boolean allowFill);
```


Indicateur d'autorisation pour le chiffrement des fichiers PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : false

```
public boolean getAllowModify();
public void setAllowModify(boolean allowModify);
```

Indicateur d'autorisation pour le chiffrement des fichiers PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : false

```
public boolean getAllowPrint();
public void setAllowPrint(boolean allowPrint);
```

Indicateur d'autorisation pour le chiffrement des fichiers PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : false

```
public boolean getAllowPrintDegraded();
public void setAllowPrintDegraded(boolean allowPrintDegraded);
```

Indicateur d'autorisation pour le chiffrement des fichiers PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : false

```
public byte[] getDecryptionKey();
public void setDecryptionKey(byte[] decryptionKey);
```

Cette option est obligatoire si le fichier PDF d'entrée est chiffré. Permet de déverrouiller la saisie sans supprimer le chiffrement. La clé du propriétaire est préférable.

Valeur par défaut : null

```
public EncryptionAlgorithm getEncryption();
public void setEncryption(EncryptionAlgorithm encryption);
```

Spécifie l'algorithme de chiffrement de la sortie.

Valeur par défaut : EncryptionAlgorithm.preserve

```
public Boolean getLinearize();
public void setLinearize(Boolean linearize);
```

Si « linearize » est défini sur « true » : la fonction de densité de probabilité (PDF) de sortie est linéarisée.

Si « linearize » est défini sur « false » : la fonction de densité de probabilité (PDF) de sortie n'est pas linéarisée.

Si « linearize » est null : la sortie n'est linéarisée que si la PDF d'entrée a été linéarisée.

Valeur par défaut : null

```
public Logger getLog();
public void setLog(Logger log);
```

Valeur par défaut : spécifiée dans le fichier de configuration.


```
public boolean getOnlyMergeXObjects();
public void setOnlyMergeXObjects(boolean onlyMergeXObjects);
```

Cette option est destinée au développement et aux tests. Il ne doit pas être utilisé dans un environnement de production.

Valeur par défaut : false

```
public byte[] getOwnerPassword();
public void setOwnerPassword(byte[] ownerPassword);
```

Le mot de passe du propriétaire pour le chiffrement du fichier PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : null

```
public PdfUaOutputHandler getStderrHandler();
public void setStderrHandler(PdfUaOutputHandler stderrHandler);
```

Sert à gérer les messages qui sont normalement affichés sur stderr. Valeur par défaut : gestionnaire intégré, tel que spécifié dans le fichier de configuration.

```
public PdfUaOutputHandler getStdoutHandler();
public void setStdoutHandler(PdfUaOutputHandler stdoutHandler);
```

Sert à gérer les messages qui sont normalement affichés sur stdout.

Valeur par défaut : gestionnaire intégré, tel que spécifié dans le fichier de configuration.

```
public PdfUaOutputHandler getTraceHandler();
public void setTraceHandler(PdfUaOutputHandler traceHandler);
```

Utilisé uniquement par la commande `<Trace>`.

Valeur par défaut : gestionnaire intégré, tel que spécifié dans le fichier de configuration.

```
public boolean getUseLatestRevisions();
public void setUseLatestRevisions(boolean useLatestRevisions);
```

Cette option est destinée au développement et aux tests. Il ne doit pas être utilisé dans un environnement de production.

Valeur par défaut : `false`

```
public byte[] getUserPassword();
public void setUserPassword(byte[] userPassword);
```

Le mot de passe utilisateur pour le chiffrement du fichier PDF. Cela n'a aucun effet si le fichier de sortie n'est pas chiffré par PDF2PDFUA.

Valeur par défaut : null

interface PdfUaOutputHandler

Package: **com.ricoh.pdf2pdfua.api**

Gestionnaire de redirection d'E/S.

```
void onMessage(Node src, int pageNo, String msg);
```

`src`: le nœud XML de la règle du fichier de contrôle qui l'appelle. Null si le message ne peut pas être associé à un nœud XML.

`pageNo`: la numérotation des pages commence à 1. Cette valeur est égale à zéro si le message ne peut pas être associé à une page.

`msg`: peut contenir plusieurs lignes de texte.

class PreprocessingException extends Exception

Package: **com.ricoh.pdf2pdfua.api**

Classe de base pour les erreurs survenant lors du prétraitement du fichier de contrôle.

class UaStackTrace extends ArrayList<String>

Package: **com.ricoh.pdf2pdfua.api**

Une trace de pile qui renvoie à un élément XML dans le fichier de contrôle.

```
public static UaStackTrace unwind(final Node top);
```

Exemples

Cet exemple montre comment analyser un fichier de contrôle et comment l'appliquer à plusieurs documents. Tous les documents d'un lot doivent être similaires et compatibles avec le fichier de contrôle. Pour des raisons de performances, le fichier de contrôle n'est analysé qu'une seule fois.

Le logiciel devrait fonctionner sur des serveurs dotés de plusieurs cœurs de processeur. Cet exemple est multithread, car il s'agit du cas d'utilisation standard.

```
package com.ricoh.pdf2pdfua.api.example;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import com.ricoh.pdf2pdfua.api.IControlFile;
import com.ricoh.pdf2pdfua.api.PDFUAException;
import com.ricoh.pdf2pdfua.api.PdfUaLoader;
import com.ricoh.pdf2pdfua.api.PdfUaOpt;
import com.ricoh.pdf2pdfua.api.Pdfua;
import com.ricoh.pdf2pdfua.api.preprocessor.PreprocessingException;
```

```
public class Pdf2PdfUaApiExample {
    static class ThreadSafeIterator {
        Iterator<File> i;

        ThreadSafeIterator(List<File> files) {
            i = files.iterator();
        }

        synchronized File next() {
            if (i.hasNext())
                return i.next();
            return null;
        }
    }

    static void exit(String msg) {
        System.err.println(msg);
        System.exit(1);
    }

    static File changeExtension(File f) {
        return new File(f.getAbsolutePath().replace(".pdf", ".ua.pdf"));
    }

    static Thread[] createThreadArray() {
        return new Thread[Runtime.getRuntime().availableProcessors()];
    }

    static void convertBatch(String controlFilePath, List<File> pdfs)
            throws PreprocessingException, InterruptedException {
        final Pdfua pdfuaInstance = PdfUaLoader.load();
        final IControlFile cf = pdfuaInstance.newControlFile(controlFilePath);
        final PdfUaOpt opt = pdfuaInstance.newOpt();
        final ThreadSafeIterator it = new ThreadSafeIterator(pdfs);
        Thread[] threads = createThreadArray();
        for (int i = 0; i < threads.length; i++) {
            threads[i] = new Thread() {
                public void run() {
                    File pdfIn;
                    while ((pdfIn = it.next()) != null) {
                        try {
                            FileInputStream fis = new FileInputStream(pdfIn);
                            File out = changeExtension(pdfIn);
                            pdfuaInstance.run(fis, out, cf, opt);
                        } catch (FileNotFoundException | PDFUAException e) {
                            e.printStackTrace();
                        }
                    }
                }
            };
        }
        for (Thread t : threads)
            t.start();
        for (Thread t : threads)
            t.join();
        System.out.println("Batch complete!");
    }

    public static void main(String[] args)
            throws PreprocessingException, InterruptedException {
        if (args.length < 1)
            exit("No control file!");
```

```
        ArrayList<File> files = new ArrayList<>();
        for (int i = 1; i < args.length; i++)
            files.add(new File(args[i]));
        if (files.isEmpty())
            exit("No PDF file to convert!");
        convertBatch(args[0], files);
    }
}
```

Cet exemple montre comment PDF2PDFUA peut fonctionner sans utiliser de fichiers. Cette fonctionnalité s'avère utile lorsque PDF2PDFUA est intégré à d'autres systèmes, tels que des applications web.

```
package com.ricoh.pdf2pdfua.api.example;

import java.io.FileInputStream;
import java.io.FileOutputStream;

import com.ricoh.pdf2pdfua.api.PdfUaLoader;
import com.ricoh.pdf2pdfua.api.Pdfua;

public class Pdf2PdfUaApiExampleBytesOutput {
    public static void main(String[] args) throws Exception {
        Pdfua ua = PdfUaLoader.load();
        byte[] outputBytes = ua.run(
            new FileInputStream("./untagged.pdf"),
            ua.newControlFile("./control-file.xml"),
            ua.newOpt()
        );
        var os = new FileOutputStream("./out.pdf");
        os.write(outputBytes);
        os.close();
    }
}
```

RICOH Transforms Suite Outil PDF2PDFUA Guide utilisateur