

IBM スポス コラボレーションおよびデプロイ
メント・サービス - パイソンのためのエッセン
シャル

8.4

開発者ガイド



注

本書および本書で紹介する製品をご使用になる前に、[71 ページの『特記事項』](#)に記載されている情報をお読みください。

<!--製品情報-->

本書は、IBM® SPSS® コラボレーションおよびデプロイメント・サービスのバージョン 8 リリース 4 モディフィケーション 0、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

© Copyright International Business Machines Corporation 2000, 2022.

目次

第 1 章 IBM スポス コラボレーションおよびデプロイメント・サービス - パイソンのためのエッセンシャル.....	1
概要.....	1
のインストール.....	1
推奨されない機能.....	1
第 2 章 コマンド・ライン・スクリプト.....	3
グローバル・キーワード.....	3
リポジトリ接続.....	4
コンテンツ・リポジトリ・スクリプト.....	5
キーワード.....	5
コンテンツ・リポジトリの操作.....	6
プロセス管理機能.....	24
キーワード.....	24
プロセス管理の操作.....	24
第 3 章 PESImpl モジュール.....	29
コンテンツ・リポジトリ API.....	30
メソッド.....	30
ラッパー・クラス.....	60
プロセス管理 API.....	63
メソッド.....	63
ラッパー・クラス.....	67
サンプル・スクリプト.....	70
特記事項.....	71
プライバシー・ポリシーに関する考慮事項.....	72
商標.....	73

第 1 章 IBM スポス コラボレーションおよびデプロイメント・サービス・パイソンのためのエッセンシャル

概要

IBM スポス コラボレーションおよびデプロイメント・サービス には、一連の API を備えたスクリプト・フレームワークが用意されています。上級ユーザーや管理者は、このフレームワークを使用して、独立したルーチンや一連のルーチンを組み合わせたバッチ・ジョブを作成し、リポジトリ・オブジェクトやリポジトリ・ジョブを処理することができます。これにより、以下のような一括処理を非常に簡単に実行することができます。

- 大規模なユーザー・グループのセキュリティー権限を変更する。
- 多数のフォルダーやファイルについて、ラベル付けやラベルの削除を行う。
- 大量のフォルダーまたはファイルのアップロードまたはダウンロード

このフレームワークには、コマンド・ラインからタスクを実行するための機能のほかに、独自の Python コード内から IBM スポス コラボレーションおよびデプロイメント・サービス・リポジトリ と対話するための豊富な API が組み込まれています。

動的なオブジェクト指向プログラミング言語である Python の一般的な情報については、[Python のサイト](#) を参照してください。

のインストール

スクリプト・フレームワークは、Windows および UNIX の各プラットフォームにインストールすることができます。スクリプト・プラットフォームは、スクリプト機能がアクセスするリポジトリで使用するプラットフォームには依存しません。

例えば、Windows プラットフォーム上で稼働するリポジトリを、UNIX プラットフォームで稼働するスクリプト機能から呼び出すことができます。

インストールについて詳しくは、「IBM スポス コラボレーションおよびデプロイメント・サービス・パイソンのためのエッセンシャル インストール・ガイド」を参照してください。

推奨されない機能

前のリリースの IBM スポス コラボレーションおよびデプロイメント・サービス からマイグレーションする場合、最後のバージョン以降に非推奨になった各種機能について注意してください。

ある機能が非推奨になった場合、IBM 会社 は、製品の今後のリリースでその機能を除去する可能性があります。将来は、推奨されるマイグレーション・アクションにリストされている戦略的機能に投資の重点が置かれます。通常、フィーチャーは、同等の代替機能が提供されない限り非推奨とはなりません。

このリリースで非推奨となったフィーチャーはありません。参照用に、製品の最近のバージョンで非推奨となったフィーチャーを以下の表に示します。この表には、可能であれば、推奨されるマイグレーション・アクションも示されます。

表 1. 以前のバージョンの非推奨になった機能	
非推奨	推奨されるマイグレーション・アクション
セキュリティー・プロバイダー: 拡張グループおよび許可ユーザーをサポートする、ローカル・オーバーライドを使用する Active Directory	標準の Active Directory セキュリティー・プロバイダーに必要なグループを追加して使用してください
IBM スポス コラボレーションおよびデプロイメント・サービスのエンタープライズ・ビュー	分析データ・ビュー機能を使用

表 1. 以前のバージョンの非推奨になった機能 (続き)	
非推奨	推奨されるマイグレーション・アクション
IBM スポス コラボレーションおよびデプロイメント・サービスのエンタープライズ・ビュー・ドライバー	分析データ・ビュー機能を使用
シナリオ・ファイル	シナリオ・ファイル(.scn) はサポートされなくなりました。エンタープライズ・ビューのソース・ノードはデプロイメント・マネージャーで変更できません。古いシナリオ・ファイルは、IBM スポス モデラー クライアントで変更し、ストリーム・ファイルとして再度保存することができます。また、シナリオ・ファイルを使用するスコアリング構成は削除し、ストリーム・ファイルに基づいて再作成する必要があります。
IBM SPSS デプロイメント・マネージャー の Web インストール	スタンドアロン・インストーラーを使用
BIRT・レポート・デザイ用の IBM スポス	なし
BIRT・レポート・デザイ用の IBM スポスビューアー	なし
IBM スポス コラボレーションおよびデプロイメント・サービス・ポートレット	IBM SPSS コラボレーションおよびデプロイメント・サービス・デプロイメント・ポータルを直接使用するか、または Web サービス API を使用してください
IBM スポス コラボレーションおよびデプロイメント・サービス ウェブ パーツ	IBM SPSS コラボレーションおよびデプロイメント・サービス・デプロイメント・ポータルを直接使用するか、または Web サービス API を使用してください
スコアリング・サービス V1 API	スコアリング・サービス V2 API
サーバー・サービスのスケジューリング	なし
レポート・サービス	なし
認証サービス login 操作	認証サービス doLogin 操作
検索サービス search 操作	検索サービス search2.5 操作
SPSS AXIS/Castor Web サービス・クライアント jar	Java ランタイム環境、統合開発環境、または Eclipse Web Tools Platform (WTP) に付属のツールを使用してください
clemrtl_setLogFile() API 関数	なし

第2章 コマンド・ライン・スクリプト

Python ファイル CADSTool.py をコマンド行から使用して、IBM スポス コラボレーションおよびデプロイメント・サービス・リポジトリ内に保管されているリソースを操作することができます。

コマンド・ラインから IBM スポス コラボレーションおよびデプロイメント・サービス スクリプト操作を呼び出すための一般的な構文は以下のとおりです。

```
python CADSTool.py <Operation> <Keywords>
```

ここで、

- <Operation> は、呼び出す機能を指定します。
- <Keywords> は、機能の入力パラメーターとして使用されるキーワードと値のペアを定義します。

グローバル・キーワード

3 ページの表 2 に、すべての IBM スポス コラボレーションおよびデプロイメント・サービス スクリプト機能でサポートされるキーワードを示します。2 列目には、オプションで使用する短縮形のキーワードを示します。キーワードは大/小文字が区別されます。

表 2. グローバル・キーワード		
キーワード	オプションの短縮形	使用法
--user	-u	リポジトリ・サーバーに接続するためのユーザー名。ネイティブ・プロバイダーからのユーザーではない場合は、値にセキュリティ・プロバイダーを表すプレフィックスが含まれていなければなりません。プレフィックスとしては以下の値が有効です。 • Native (システム固有のネイティブ・ローカル・セキュリティ・プロバイダーの場合)。これはデフォルト・プロバイダーです。 • AD_<name> for Active Directory。ここで、<name>は、システム内のセキュリティ・プロバイダー名に対応します。 • ADL_<name> for Active Directory (ローカル・オーバーライド)。ここで、<name>は、システム内のセキュリティ・プロバイダー名に対応します。 • ldap_<name> (OpenLDAP の場合)。ここで、<name>は、システム内のセキュリティ・プロバイダー名に対応します。 プレフィックスの後ろにスラッシュを付け、その後にユーザー名を続けます。Active Directory プロバイダーの場合は、プレフィックスにドメインを含めます。例えば、Active Directory インスタンス AD_SPSSAD の MYDOMAIN ドメイン内のユーザー icrod の場合、値は AD_SPSSAD/MYDOMAIN/icrod となります。ユーザー icrod が OpenLDAP プロバイダー SPSSLDAP 内にある場合は、値は ldap_SPSSLDAP/icrod となります。
--password	-p	リポジトリ・サーバーに接続するためのパスワード。暗号化されたパスワードがサポートされます。

表 2. グローバル・キーワード (続き)		
キーワード	オプションの短縮形	使用法
--host	-q	リポジトリがインストールされているホスト/サーバー名
--port	-o	リポジトリ・サーバーのポート番号
--ssl		リポジトリ・サーバーが Secure Sockets Layer (SSL) プロトコルを使用して通信を暗号化することを指定します。このキーワードを使用する場合は、リポジトリ・サーバーを SSL 用に構成する必要があります。詳しくは、管理者用の資料を参照してください。
--server_url	-S	リポジトリ・サーバーの完全な URL。サーバーの URL にカスタムのコンテキスト・ルートが含まれている場合は、このキーワードを使用してください。サーバーの URL を指定する場合は、host、port、および ssl の各キーワードに値を指定する必要はありません。
--appserv_type		リポジトリ・サーバーのデプロイ対象のアプリケーション・サーバー・タイプを示します。
--useDefault	-z	Authorization.properties ファイルで定義されているサーバー接続情報を使用します。
--help	-h	スクリプト・モジュールのヘルプ情報

リポジトリ接続

すべてのコマンドの末尾に、IBM スポス コラボレーションおよびデプロイメント・サービス・リポジトリのユーザー ID、パスワード、およびリポジトリ・サーバーの情報を指定する必要があります。

以下の方法で、この接続情報を指定できます。

- 方法 1: キーワードを使用する。以下に例を示します。

```
--user user --password password --appserv_type appservtype --host host --port port
```

```
--user user --password password --appserv_type appservtype --server_url url
```

- 方法 2: コマンドに --useDefault パラメーター (または短縮形の -z) が指定された Authorization.properties ファイルを使用する。この方法では、*Scripting folder\Lib\site-packages\config\Authorization.properties* にある Authorization.properties ファイルから接続情報を取得します。リポジトリの設定と一致するように、一般的なテキスト・エディターを使用してファイル内の以下の値を変更します。

```
# Authorization Information
user=admin
password=pwd
host=yourhost
port=80
appserv_type=websphere
```

または、プロパティ・ファイル内の server_url キーワードを使用できます。

```
# Authorization Information
user=admin
password=pwd
server_url=http://yourhost:80/context_root
appserv_type=websphere
```

コマンド・ラインで指定したパラメーターが常に優先されます。例えば、コマンド・ラインで --user と --password を指定し、さらに --useDefault パラメーター (または -z パラメーター) を指定した場合は、コマンド・ラインで指定したユーザーとパスワードが使用され、ホストとポートは

Authorization.properties ファイルから取得されます。また、ユーザー、パスワード、ホスト、ポートをすべてコマンド・ラインで指定し、--useDefault パラメーター (または -z パラメーター) も指定した場合は、--useDefault が無視され、コマンド・ラインで指定した情報だけが使用されます。

指定するパスワードを暗号化できます。パスワード値を暗号化するには、管理者に連絡してください。管理者は、-portable オプションを指定して repository_server_installation_directory/bin/cliEncrypt ユーティリティを実行し、パスワードを暗号化できます。

```
repository_server_installation_directory/bin/cliEncrypt password -portable
```

ここで説明するすべての API の構文と例について、-z パラメーターを使用して、必須パラメーターの使用数を最小限に抑えています。

コンテンツ・リポジトリ・スクリプト

コンテンツ・リポジトリ・スクリプトには、ファイルやフォルダーなどのリポジトリ・リソースを処理する機能が組み込まれています。以下の機能を備えています。

- フォルダーの作成および削除
- ファイルのアップロードおよびダウンロード
- フォルダーのエクスポートおよびインポート
- ラベル、セキュリティ、およびメタデータの管理

このセクションでは、リポジトリ機能用のスクリプトを Python のコマンド・ラインで使用方法について説明します。すべての操作について、詳細な構文情報、例、予測されるメッセージを記載しています。

キーワード

5 ページの表 3 に、リポジトリ機能でサポートされるキーワードを示します。2 列目には、オプションで利用できる短縮形のキーワードを示します。

重要: キーワードは大/小文字が区別されます。

表 3. リポジトリ API のキーワード		
キーワード	オプションの短縮形	使用法
--source	-s	ソース・ファイルまたはフォルダーのパス
--target	-t	ターゲット・フォルダー・パス
--version	-v	ファイルのバージョン
--principal	-r	許可を付与する必要がある対象のユーザー
--permission	-n	許可のタイプ (読み取り、書き込み、変更、削除など)
--label	-l	ファイルのバージョンに割り当てるラベル
--criteria	-c	ファイルまたはフォルダーのメタデータ属性を検索するための検索条件。
--author	-a	ファイルまたはフォルダーの作成者の名前
--description	-d	ファイルまたはフォルダーの説明
--title	-i	ファイルまたはフォルダーのタイトル
--expirationDate	-q	ファイルまたはフォルダーの有効期限
--expirationStartDate		ファイルまたはフォルダーの有効期間の開始日
--expirationEndDate		ファイルまたはフォルダーの有効期間の終了日

表 3. リポジトリ API のキーワード (続き)		
キーワード	オプションの短縮形	使用法
--keyword	-k	ファイルまたはフォルダーのキーワード
--cascade	-x	フォルダーのセキュリティ設定をサブフォルダーとファイルに適用することを指定します。
--provider	-f	プリンシパルを取得するために使用するセキュリティ・プロバイダー
--createVersion	-b	ファイルの新しいバージョンを作成することを指定します。
--contentLanguage	-g	ファイルまたはフォルダーのコンテンツ言語
--topic		ファイルまたはフォルダーに割り当てるトピック。複数の値を入力することができます (--topic "topic1;topic2" など)
--modifiedBy		ファイルまたはフォルダーを変更したユーザー
--mimeType		ファイルの MIME タイプ
--createdBy		ファイルまたはフォルダーを作成したユーザー
--submittedHierarchy		「Submitted Jobs」フォルダーを検索するかどうかを指定します。
--propertyName		カスタム・プロパティの名前
--customProperty		更新するカスタム・プロパティの名前と値のペア。
--propertyName		有効な値を取得するカスタム・プロパティの名前

ラベル情報とバージョン情報を受け入れる操作の場合は、ラベルとバージョンのいずれか一方だけを指定してください。両方を指定することはできません。対象ファイルのバージョンとラベルを指定しなかった場合は、最新のバージョンが使用されます。

コンテンツ・リポジトリの操作

advanceSearch 操作

さまざまなパラメーターに基づき、リポジトリでファイルやフォルダーを検索します。

構文

```
python CADSTool.py advanceSearch --author <author>
--title <title> --description <description>
--createdBy <createdBy> --modifiedBy <modifiedBy>
--keyword <keyword> --label <label>
--topic <topic>
--uri <uri> --parentURI <parentURI>
--expirationStartDate <expirationStartDate>
--expirationEndDate <expirationEndDate>
--createdStartDate <createdStartDate>
--createdEndDate <createdEndDate>
--objectModifiedStartDate <objectModifiedStartDate>
--objectModifiedEndDate <objectModifiedStartDate>
--versionModifiedStartDate <versionModifiedStartDate>
--versionModifiedEndDate <versionModifiedEndDate>
--submittedHierarchy -z
```

ここで、

- <author>は作成者の名前です。
- <title>は、ファイル/フォルダーのタイトルです。

例

```
python CADSTool.py advanceSearch --label "Production" --keyword "Quarterly"
--useDefault -z
```

```
python CADSTool.py advanceSearch --createdStartDate "2009-12-01T00:00:00+01:00"
--createdEndDate "2010-12-15T21:33:40+01:00" -z
```

```
python CADSTool.py advanceSearch --uri "spsscr:///id=a010a37ba5992bb00000127b0f952f945be" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- API が正常に完了した場合は、検索条件に一致するすべてのファイルとフォルダーのリストが表示されます。通常、このリストには、完全修飾パス付きのファイル名とバージョンが表示されます。
- Error searching files and folders
- error: no such option:<option>

applySecurity 操作

リポジトリ内のファイルまたはフォルダーに対するセキュリティ・アクセス・コントロール・リスト (ACL) を設定します。

構文

```
python CADSTool.py applySecurity --source "<source>" --principal "<principal>"
--permission "<permission>" --provider "<provider>" --cascade -z
```

ここで、

- **<source>**は、セキュリティ ACL を適用するファイルまたはフォルダーの完全修飾 IBM スポス コラボレーションおよびデプロイメント・サービス・リポジトリパスです。これは必須パラメーターです。
- **<principal>**は、ACL の一部として指定のファイルまたはフォルダーに適用するユーザー (管理者など) です。これは必須パラメーターです。
- **<permission>**は、指定されたファイルまたはフォルダーに適用する許可のタイプ (読み取り、書き込み、変更、削除、所有者など) です。これは必須パラメーターです。
- **<provider>**は、ユーザー (プリンシパル) に関する情報を取得するために使用するセキュリティ・プロバイダーです。これはオプション・パラメーターです。有効な値は次のとおりです。
 - Native (システム固有のネイティブ・ローカル・セキュリティ・プロバイダーの場合)。これはデフォルト・プロバイダーです。
 - AD_<name> for Active Directory。ここで、<name>は、システム内のセキュリティ・プロバイダー名に対応します。
 - ADL_<name> for Active Directory (ローカル・オーバーライド)。ここで、<name>は、システム内のセキュリティ・プロバイダー名に対応します。
 - ldap_<name> (OpenLDAP の場合)。ここで、<name>は、システム内のセキュリティ・プロバイダー名に対応します。
- **--cascade** は、フォルダーにセキュリティを設定し、そのセキュリティ設定を指定のフォルダー内のすべてのファイルとサブフォルダーに適用する場合に使用します。これはオプション・パラメーターです。

例

以下の例では、フォルダーにセキュリティを適用します。

```
python CADSTool.py applySecurity --source "/Projects" --principal "icrod"
--permission "READ" --provider "Native" -z
```

以下の例では、フォルダーと、そのフォルダー内のすべてのファイルとサブフォルダーにセキュリティを適用します。

```
python CADSTool.py applySecurity --source "/Projects/" --principal "icrod"
--permission "READ" --provider "Native" --cascade -z
```

以下の例は、SPSSAD という名前の Active Directory セキュリティー・プロバイダー内のプリンシパルのフォルダーにセキュリティを適用します。

```
python CADSTool.py applySecurity --source "/Projects" --principal "ICrod (MYDOMAIN)"
--permission "Write" --provider "AD_SPSSAD" -z
```

以下の例は、LDAP という名前の OpenLDAP セキュリティー・プロバイダー内のプリンシパルのフォルダーにセキュリティを適用します。

```
python CADSTool.py applySecurity --source "/Projects" --principal "ICrod (LDAP)"
--permission "Read" --provider "ldap_LDAP" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <permission> permission granted successfully to <principal> on <source>.
- <source> does not exist.
- <permission> should be one of ['Read', 'Delete', 'Modify ACL', 'owner'].
- <source> Error setting security ACL.

cascadeSecurity 操作

フォルダーのセキュリティ設定を、そのフォルダー内のすべてのファイルとサブフォルダーに適用します。

構文

```
python CADSTool.py cascadeSecurity --source "<source>" -z
```

<source>の値は、リポジトリ内のフォルダーの完全修飾パスです。これは必須パラメーターです。

例

```
python CADSTool.py cascadeSecurity --source "/Projects" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- Security ACL cascaded successfully for <source>.
- <source> does not exist.
- <source> Error cascading security ACL.

copyResource 操作

ファイルまたはフォルダーをリポジトリ内の別のフォルダーにコピーします。

この API には名前変更機能が組み込まれているため、指定したファイルをコピーする際に名前を変更することができます。18 ページの『[moveResource 操作](#)』の先頭で説明されているケースは、この copyResource API にも適用されます。

構文

```
python CADSTool.py copyResource --source "<source>" --target "<target>" -z
```

ここで、

- **<source>**は、コピーするファイル/フォルダーの完全修飾コンテンツ・リポジトリ・パスです。これは必須パラメーターです。
- **<target>**は、ファイル/フォルダーがコピーされるリポジトリの完全修飾パスです。これは必須パラメーターです。

例

以下の例では、ファイルをコピーします。

```
python CADSTool.py copyResource --source "/Demo/Drafts/MyReport.rptdesign" --target  
"/Projects" -z
```

以下の例では、ファイルをコピーしてファイル名を変更します。

```
python CADSTool.py copyResource --source "/Demo/Drafts/MyReport.rptdesign" --target  
"/Projects/Report.rptdesign" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- **<source> copied successfully to <target>.**
- **<source> does not exist.**
- **<target> resource not found.**
- **<source> Error copying file or folder.**

createFolder 操作

リポジトリ内の指定した場所に新しいフォルダーを作成します。

構文

```
python CADSTool.py createFolder --source "<source>" -z
```

<source>の値は、作成する新規フォルダーの完全修飾パスです。これは必須パラメーターです。指定されたパスに基づき、新しいフォルダーが(サブフォルダーが指定された場合はそのサブフォルダーも含めて)作成されます。

例

以下の例では、*Drafts* フォルダーがまだ作成されていない場合に、このフォルダーを作成します。

```
python CADSTool.py createFolder --source "/Demo/Drafts" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- **<source> Folder created successfully.**
- **<folder> already exists.**
- **<source> Error creating folder.**

deleteFile 操作

ファイルとそのファイルのすべてのバージョンをリポジトリから削除します。

構文

```
python CADSTool.py deleteFile --source "<source>" --submittedHierarchy -z
```

ここで、

- **<source>**は、削除するファイルのリポジトリの完全修飾パスです。これは必須パラメーターです。
- **--submittedHierarchy** は、ファイルを「Submitted Jobs」フォルダーから削除します。これはオプション・パラメーターです。

例

以下の例では、ファイルとそのファイルのすべてのバージョンをリポジトリから削除します。

```
python CADSTool.py deleteFile --source "/Demo/Drafts/MyReport.rptdesign" -z
```

以下の例では、ファイルとそのファイルのすべてのバージョンを「Submitted Jobs」フォルダーから削除します。

```
python CADSTool.py deleteFile --source "Submitted Jobs/admin/  
2007-05-21.14.10.22.422-test.dbq/test.dbq.html" --submittedHierarchy -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- **<source> deleted successfully.**
- **<source> does not exist.**
- **<source> Error deleting file.**

deleteFileVersion 操作

リポジトリから、ファイルの特定のバージョンを削除します。

構文

```
python CADSTool.py deleteFileVersion --source "<source>" --version "<version>"  
--label "<label>" --submittedHierarchy -z
```

ここで、

- **<source>**は、削除するファイルのリポジトリの完全修飾パスです。これは必須パラメーターです。
- **<version>**は、削除するファイルの特定のバージョンです。これはオプション・パラメーターです。
- **<label>**は、削除するファイルのラベルです。これはオプション・パラメーターです。
- **--submittedHierarchy** は、ファイルの特定のバージョンを「Submitted Jobs」フォルダーから削除します。これはオプション・パラメーターです。

例

以下の例では、ファイルの特定のバージョンを削除します。

```
python CADSTool.py deleteFileVersion --source "/Demo/Drafts/MyReport.rptdesign" --version  
"0:2006-08-25 21:15:49.453" -z
```

以下の例では、特定のラベルが付いたファイルを削除します。

```
python CADSTool.py deleteFileVersion --source "/Demo/Drafts/MyReport.rptdesign" --label "Test" -z
```

以下の例では、特定のラベルが付いたファイルを「Submitted Jobs」フォルダーから削除します。

```
python CADSTool.py deleteFileVersion --source "Submitted Jobs/admin/2007-05-21.14.10.22.422-test.dbq/test.dbq.html" --label "Test" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> with version/label <version>/<label> deleted.
- <source> does not exist.
- Either Label or Version can be specified.
- Version or Label is required.
- <source> Error deleting file.

deleteFolder 操作

フォルダーとそのフォルダー内のすべての内容をリポジトリから削除します。

構文

```
python CADSTool.py deleteFolder --source <source> --submittedHierarchy -z
```

ここで、

- <source>は、削除するフォルダーのリポジトリの完全修飾パスです。これは必須パラメーターです。
- --submittedHierarchy は、フォルダーの特定のバージョンを「Submitted Jobs」フォルダーから削除します。これはオプション・パラメーターです。

例

以下の例では、フォルダーを削除します。

```
python CADSTool.py deleteFolder --source "/Demo/Drafts" -z
```

以下の例では、フォルダーを「Submitted Jobs」フォルダーから削除します。

```
python CADSTool.py deleteFolder --source "Submitted Jobs/admin/2007-05-21.14.10.22.422-test.dbq/" --submittedHierarchy -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> deleted successfully.
- <source> does not exist.
- <source> Error deleting folder.

downloadFile 操作

リポジトリ内の特定のバージョンのファイルをローカル・ファイル・システムにダウンロードします。

構文

```
python CADSTool.py downloadFile --source "<source>" --version "<version>" --label "<label>" --target "<target>" -z
```

ここで、

- **<source>**は、リポジトリの完全修飾パスです。これは必須パラメーターです。
- **<version>**は、ダウンロードするファイルのバージョンです。これはオプション・パラメーターです。
- **<label>**は、ダウンロードするファイルのラベルです。これはオプション・パラメーターです。
- **<target>**は、ファイルがダウンロードされる (ローカル・ファイル・システム上の) 完全修飾パスです。

例

以下の例では、ファイルの最新バージョンをダウンロードします。

```
python CADSTool.py downloadFile --source "/Demo/Drafts/MyReport.rptdesign"
--target "C:/Demo/Shared/" -z
```

以下の例では、バージョン・マーカを使用してファイルの特定のバージョンをダウンロードします。

```
python CADSTool.py downloadFile --source "/Demo/Drafts/MyReport.rptdesign" --version
"0:2006-08-25 21:15:49.453" --target "C:/Demo/Shared/" -z
```

以下の例では、ラベル付きバージョンのファイルをダウンロードします。

```
python CADSTool.py downloadFile --source "/Demo/Drafts/MyReport.rptdesign" --label "Production"
--target "C:/Demo/Shared/" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- **<source>** downloaded successfully.
- **<source>** with label **<label>** downloaded successfully.
- **<source>** with version **<version>** downloaded successfully.
- **<source>** with version **<version>** does not exist.
- **<source>** with label **<label>** does not exist.
- **<target>** does not exist.
- **<source>** Error downloading File.

export 操作

コンテンツ・リポジトリからのエクスポートを開始します。この操作では、エクスポートするファイルとフォルダーを選択して、*.pes エクスポート・ファイルをローカル・ファイル・システムに保存することができます。

構文

```
python CADSTool.py export --source "<source>" --target "<target>" -z
```

ここで、

- **<source>**は、エクスポートするフォルダーのリポジトリの完全修飾パスです。これは必須パラメーターです。
- **<target>**は、作成する *.pes エクスポート・ファイルの (ローカル・ファイル・システム上の) 完全修飾パスです。これは必須パラメーターです。

例

```
python CADSTool.py export --source "/Projects/" --target "C:\Demo\drafts.pes" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> exported successfully to <target>.
- <source> does not exist.
- <source> Error exporting folder.
- No such file or directory: <target>

getAccessControlList 操作

コンテンツ・リポジトリ内の指定されたファイルまたはフォルダーのセキュリティー・アクセス・コントロール・リスト (ACL) を取得します。

構文

```
python CADSTool.py getAccessControlList --source "<source>" -z
```

<source>の値は、ファイル/フォルダーの完全修飾パスです。これは必須パラメーターです。

例

```
python CADSTool.py getAccessControlList --source "/Projects/MyReport.rptdesign" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> does not exist.
- Error retrieving security details for <source>.

getAllVersions 操作

リポジトリ内のファイルのすべてのバージョンのリストを取得します。

構文

```
python CADSTool.py getAllVersions --source "<source>" --submittedHierarchy -z
```

ここで、

- <source>は、バージョンを取得するファイルのリポジトリの完全修飾パスです。これは必須パラメーターです。
- --submittedHierarchy は、バージョンを「Submitted Jobs」フォルダーから取得します。これはオプション・パラメーターです。

例

以下の例では、指定されたファイルのすべてのバージョンを取得します。

```
python CADSTool.py getAllVersions --source "/Demo/Drafts/MyReport.rptdesign" -z
```

以下の例では、指定されたファイルのすべてのバージョンを「Submitted Jobs」フォルダーから取得します。

```
python CADSTool.py getAllVersions --source "Submitted Jobs/admin/2007-05-21.14.10.22.422-test.dbq/test.dbq.html" --submittedHierarchy -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> does not exist.
- <source> Error retrieving file versions.

処理が正常に終了すると、バージョン・マーカータラベル情報など、ファイル・バージョンの情報がすべて表示されます。

getChildren 操作

リポジトリで指定されたフォルダー内のすべてのファイルとフォルダーのリストを取得します。

構文

```
python CADSTool.py getChildren --source "<source>" -z
```

<source>の値は、フォルダーの完全修飾パスです。これは必須パラメーターです。

例

```
python CADSTool.py getChildren --source "/Demo/Drafts" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> does not exist.
- <source> Error getting resources.

getCustomPropertyValue 操作

指定されたカスタム・プロパティで利用できる有効な値を取得します。

構文

```
python CADSTool.py getCustomPropertyValue --propertyName "<propertyName>" -z
```

<propertyName>の値は、カスタム・プロパティの名前です。これはオプション・パラメーターです。

例

```
python CADSTool.py getCustomPropertyValue --propertyName "Language" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <propertyName> takes values as <valid values>
- Error getting property details for <propertyName>
- 'Custom Property Name not specified'

- Error retrieving property details for <propertyName>.

getMetadata 操作

リポジトリ内のファイルまたはフォルダーのメタデータ属性を取得します。

構文

```
python CADSTool.py getMetadata --source "<source>" --version "<version>" --label  
"<label>" --submittedHierarchy -z
```

ここで、

- <source>は、メタデータを取得する対象のファイルまたはフォルダーの完全修飾リポジトリ・パスです。フォルダーの場合、version 属性と label 属性は無視されます。これは必須パラメーターです。
- <version>は、メタデータを取得するファイルのバージョンです。これはオプション・パラメーターです。
- <label>は、メタデータを取得するファイルのラベルです。これはオプション・パラメーターです。
- --submittedHierarchy は、メタデータを「Submitted Jobs」フォルダーから取得します。これはオプション・パラメーターです。

例

以下の例では、フォルダーのメタデータを取得します。

```
python CADSTool.py getMetadata --source "/Demo/Drafts" -z
```

以下の例では、ラベル付きバージョンのファイルのメタデータを取得します。

```
python CADSTool.py getMetadata --source "/Demo/Drafts/MyReport.rptdesign" --label "Test" -z
```

以下の例では、「Submitted Jobs」フォルダー内のラベル付きバージョンのファイルのメタデータを取得します。

```
python CADSTool.py getMetadata --source "Submitted Jobs/admin/  
2007-05-21.14.10.22.422-test.dbq/test.dbq.html" --label "LATEST" --submittedHierarchy -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> does not exist.
- 'Either Label or Version can be specified'
- <source> Error retrieving file metadata.
- 処理が正常に終了すると、カスタムのメタデータ・プロパティーも含めて、指定したファイルまたはフォルダーのメタデータ情報がすべて表示されます。

import 操作

ローカル・ファイル・システム上の既存の *.pes エクスポート・ファイルをリポジトリにインポートします。

構文

```
python CADSTool.py import --source "<source>" --target "<target>"  
--resourceType "<type>"  
--resourceConflict "<rconflict>"  
--labelFrom "<label>"  
--lockResolution "<resolution>"  
--invalidVersionConflict "<vconflict>"  
--resourceDef "<rdefinition>"
```

```
--exclude "<exclusion>"
-z
```

ここで、

- **<source>**は、リポジトリにインポートする*.pes エクスポート・ファイルの(ローカル・ファイル・システム上の)完全修飾パスです。これは必須パラメーターです。
- **<target>**は、*.pes エクスポート・ファイルのインポート先のリポジトリの完全修飾パスです。これは必須パラメーターです。
- **<type>**は、インポートされるコンテンツのタイプを示します。次のいずれかの値を指定します。
 - **ContentRepository**: ファイルやフォルダーなどのコンテンツ・オブジェクトの場合
 - **ResourceDef**: リソース定義の場合
 - **Credential**: ユーザー資格情報の場合
 - **DataSource**: データ・ソース定義の場合
 - **MessageDomain**: メッセージ・ドメインの場合
 - **ServerCluster**: サーバー・クラスター定義の場合
 - **Server**: サーバー定義の場合
 - **PromotionPolicy**: プロモーション・ポリシーの場合

これはオプション・パラメーターです。このパラメーターを指定しない場合は、デフォルト値の **ContentRepository** が使用されます。

- **<rconflict>**は、重複する ID または名前の競合を解決する方法を示します。次のいずれかの値を指定します。
 - **keepTarget**. ターゲット項目が保持されます。*.pes ファイルに含まれる、ID が重複しているソース項目は、無視されます。
 - **addNewVersion**. 通常、このオプションを使用すると ID または名前の競合が解決されます。ソース・オブジェクトとターゲット・オブジェクトの間で重複した ID の競合が発生した場合、新しいバージョンのオブジェクトが対象の場所に作成されます。名前の競合が発生した場合、対象の場所にあるインポートされたオブジェクトの名前が変更されます。通常、名前の変更が行われたオブジェクトには、_1、_2、などが付加されます。2つのバージョンのオブジェクトに同じラベルが付いている場合、2つのバージョンの同じ項目に同じラベルを使用することはできないため、システムは一方のラベルを保持し、重複したラベルを破棄します。保持されるラベルは、**labelFrom** パラメーターによって異なります。

これはオプション・パラメーターです。このパラメーターを指定しない場合は、デフォルト値の **keepTarget** が使用されます。

- **<label>**は、オブジェクトの2つのバージョンが同じラベルを持つ場合に使用するラベルを指定します。他方のバージョンのラベルは破棄されます。次のいずれかの値を指定します。
 - **source**
 - **target**

これはオプション・パラメーターです。このパラメーターを指定しない場合は、デフォルト値の **source** が使用されます。

- **<resolution>**は、ロックされたリソースが検出された場合の処理方法を定義します。次のいずれかの値を指定します。
 - **continue** ロックされたリソースを省略して、インポートを続行します。
 - **abort** ロックされたリソースが検出されると、インポート・プロセスを終了します。競合がオブジェクトのロックにより発生した場合は、インポート処理は終了され、失敗します。

これはオプション・パラメーターです。このパラメーターを指定しない場合は、デフォルト値の **abort** が使用されます。

- **<vconflict>**は、インポート・プロセス中に無効なバージョンが検出された場合の処理方法を定義します。次のいずれかの値を指定します。
 - **import** 無効なバージョンがインポートされます。
 - **discard** 無効なバージョンは削除されます。
 これはオプション・パラメーターです。このパラメーターを指定しない場合は、デフォルト値の **import** が使用されます。
- **<rdefinition>**は、リソース定義の処理動作を定義します。次のいずれかの値を指定します。
 - **recommended**. リソース定義は、識別子または名前が対象の定義と競合しない場合にのみインポートされます。競合しているリソース定義はいずれもインポートされません。
 - **include**. インポート・ファイル内のすべてのリソース定義がインポートされます。対応するチェック・ボックスを選択して、インポートから除外する 1 つ以上のリソース定義の種類を選択することができます。
 - **exclude**. インポート・ファイルからのリソース定義はインポートされません。インポートされたオブジェクトは、使用可能なリソース定義を参照するように変更する必要があります。
 これはオプション・パラメーターです。このパラメーターを指定しない場合は、デフォルト値の **recommended** が使用されます。
- **<exclusion>**は、インポート時に除外するリソース・タイプを定義します。複数の値を任意の順にセミコロンで区切って結合できます。以下の値を 1 つ以上指定します。
 - **credential**: ユーザー資格情報を除外します
 - **customproperty**: リソース・オブジェクトのカスタム・プロパティを除外します
 - **datasource**: データ・ソース定義を除外します
 - **messagedomain**: メッセージ・ドメインを除外します
 - **notification**: 通知定義を除外します
 - **servercluster**: サーバー・クラスター定義を除外します
 - **server**: サーバー定義を除外します
 - **topic**: トピックの定義を除外します
 これはオプション・パラメーターです。このパラメーターを指定しない場合は、すべてのタイプがインポートに含まれます。

例

```
python CADSTool.py import --source "C:\Demo\drafts.pes" --target "/Demo/Drafts/"
--resourceConflict "addNewVersion" --labelFrom "target" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- **<source> imported successfully to <target>.**
- **<source> does not exist.**
- **<target> does not exist.**
- **<source> Error importing folder.**

moveResource 操作

ファイルまたはフォルダーをリポジトリ内の別のフォルダーに移動します。

この API には名前変更機能が組み込まれているため、指定したファイルやフォルダーを移動する際に名前を変更することができます。以下の例を使用して、名前変更機能の動作について説明します。

ソースが */Temp Folder/Temp.txt* で、ターゲットが */Demo Folder* の場合、以下の動作になります。

- **ケース 1:** フォルダー *Demo Folder* が存在する場合は、*Temp.txt* が *Demo Folder* に移動されます。
- **ケース 2:** フォルダー *Demo Folder* が存在しない場合、*Temp.txt* は */* に移動され、*Demo Folder* に名前変更されます。

ソースが */Temp Folder/Temp.txt* で、ターゲットが */Demo Folder/Abc.dat* の場合、以下の動作になります。

- **ケース 1:** フォルダー *Demo Folder* が存在する場合は、*Temp.txt* が *Demo Folder* に移動され、名前が *Abc.dat* に変更されます。
- **ケース 2:** フォルダー *Demo Folder* が存在しない場合は、エラーが表示されます。

構文

```
python CADSTool.py moveResource --source "<source>" --target "<target>" -z
```

ここで、

- **<source>**は、移動するファイルまたはフォルダーのリポジトリの完全修飾パスです。これは必須パラメーターです。
- **<target>**は、ファイルまたはフォルダーが移動されるリポジトリの完全修飾パスです。これは必須パラメーターです。

例

以下の例では、ファイルを移動します。

```
python CADSTool.py moveResource --source "/Demo/Drafts/MyReport.rptdesign" --target  
"/Approved" -z
```

以下の例では、フォルダーを移動します。

```
python CADSTool.py moveResource --source "/Demo/Drafts/" --target "/Projects" -z
```

以下の例では、ファイルを移動してファイル名を変更します。

```
python CADSTool.py moveResource --source "/Demo/Drafts/MyReport.rptdesign" --target  
"/Approved/Report.rptdesign" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- **<source>** moved successfully to **<target>**.
- **<source>** does not exist.
- **<target>** does not exist.
- **<source>** Error moving file or folder.

removeLabel 操作

リポジトリ内のファイルからラベルを削除します。

構文

```
python CADSTool.py removeLabel --source "<source>" --label "<label>" -z
```

ここで、

- **<source>**は、リポジトリ内のファイルの完全修飾パスです。これは必須パラメーターです。
- **<label>**は、指定されたファイルから削除するラベル名です。これは必須パラメーターです。

例

```
python CADSTool.py removeLabel --source "/Demo/Drafts/MyReport.rptdesign"  
--label "Draft" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- Label removed successfully for <source>.
- Label not specified. Please try again.
- <source> with label <label> does not exist.
- <source> Error deleting label.

removeSecurity 操作

リポジトリ内の指定されたファイルまたはフォルダーからセキュリティー・アクセス・コントロール・リスト (ACL) を削除します。

構文

```
python CADSTool.py removeSecurity --source "<source>" --principal "<principal>"  
--provider "<provider>" --cascade -z
```

ここで、

- <source>は、セキュリティーを削除するファイル/フォルダーの完全修飾パスです。これは必須パラメーターです。
- <principal>は、指定されたファイル/フォルダーからセキュリティーを除去するユーザー/プリンシパル (管理者など) です。これは必須パラメーターです。
- <provider>は、ユーザー (プリンシパル) に関する情報を取得するために使用するセキュリティー・プロバイダーです。これはオプション・パラメーターです。有効な値は次のとおりです。
 - Native (システム固有のネイティブ・ローカル・セキュリティー・プロバイダーの場合)。これはデフォルト・プロバイダーです。
 - AD_<name> for Active Directory。ここで、<name>は、システム内のセキュリティー・プロバイダー名に対応します。
 - ADL_<name> for Active Directory (ローカル・オーバーライド)。ここで、<name>は、システム内のセキュリティー・プロバイダー名に対応します。
 - ldap_<name> (OpenLDAP の場合)。ここで、<name>は、システム内のセキュリティー・プロバイダー名に対応します。
- --cascade は、フォルダーからセキュリティーを削除し、そのセキュリティー設定を指定のフォルダー内のすべてのファイルとサブフォルダーから削除する場合に使用します。これはオプション・パラメーターです。

例

```
python CADSTool.py removeSecurity --source "/Projects/MyReport.rptdesign"  
--principal "icrod" --provider "Native" --cascade -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- Removed permission granted to <principal> to <source> successfully.
- <source> does not exist.
- Invalid principal.

- <source> Error deleting security ACL.

search 操作

リポジトリ内でファイルやフォルダーを検索します。検索結果として、検索条件に一致するファイルやフォルダーとそのバージョンのリストを返します。

構文

```
python CADSTool.py search --criteria "<criteria>" -z
```

<criteria>の値は、リポジトリ内のすべてのファイルおよびフォルダーのメタデータを検索するために使用される検索ストリングです。これは必須パラメーターです。

例

```
python CADSTool.py search --criteria "Quarterly" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- 検索が正常に完了した場合は、検索条件に一致するすべてのファイルとフォルダーのリストが表示されます。通常、このリストには、完全修飾パス付きのファイル名とバージョンが表示されます。
- Error searching files and folders.

setLabel 操作

リポジトリ内の任意のバージョンのファイルにラベルを適用します。既にファイルにラベルが付いている場合は、元のラベルを削除して新しいラベルで置き換えます。

構文

```
python CADSTool.py setLabel --source "<source>" --version "<version>" --label  
"<label>" -z
```

ここで、

- <source>は、リポジトリ内のファイルの完全修飾パスです。これは必須パラメーターです。
- <version>は、ラベルを適用するファイルのバージョンです。これは必須パラメーターです。
- <label>は、指定されたバージョンのファイルに適用するラベル名です。これは必須パラメーターです。

例

```
python CADSTool.py setLabel --source "/Demo/Drafts/MyReport.rptdesign" --version  
"1:2006-08-25 21:15:49.453" --label "Beta" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- Label <label> set successfully for <source> with version <version>.
- <source> does not exist.
- <source> Error setting label.

setMetadata 操作

メタデータ・プロパティをリポジトリ内のファイルとフォルダーに適用します。

22 ページの表 4 にメタデータ・プロパティーを示します。また、これらのプロパティーをファイルとフォルダーに適用できるかどうかを示します。

表 4. メタデータ・プロパティーとリソース・タイプ	
メタデータ・プロパティー	リソース・タイプ
作成者	ファイル
説明	ファイルまたはフォルダー
タイトル	ファイルまたはフォルダー
有効期限	ファイルまたはフォルダー
キーワード	ファイル
トピック	ファイル
カスタム・メタデータ	ファイルまたはフォルダー

構文

```
python CADSTool.py setMetadata --source "<source>" --version "<version>" --label
"<label>" --author "<author>" --title "<title>" --description "<description>"
--expirationDate "<expirationDate>" --topic "<topic>" --keyword "<keyword>"
--customProperty "<customProperty>" -z
```

ここで、

- **<source>**は、メタデータを設定するファイルまたはフォルダーの完全修飾リポジトリ・パスです。これは必須パラメーターです。
- **<author>**は、ファイルまたはフォルダーの作成者です。これはオプション・パラメーターです。
- **<title>**は、ファイルまたはフォルダーのタイトルです。これはオプション・パラメーターです。
- **<description>**は、ファイル/フォルダーの説明です。これはオプション・パラメーターです。
- **<expirationDate>**は、ファイルまたはフォルダーの有効期限です。これはオプション・パラメーターです。日付形式は、YYYY-MM-DDThh:mm:ssTZD です (例: 1997-07-16T19:20:30+01:00)。各部の意味を以下に示します。

YYYY: 4 桁の年

MM: 2 桁の月 (01 が 1 月。以下 12 月まで同様)

DD: 2 桁の日 (01 から 31 まで)

hh: 2 桁の時間 (00 から 23 まで。am/pm なし)

mm: 2 桁の分 (00 から 59 まで)

ss: 2 桁の秒 (00 から 59 まで)

TZD: タイム・ゾーン指定子 (Z または +hh:mm または -hh:mm)

- **<keyword>**は、ファイルまたはフォルダーのキーワードです。これはオプション・パラメーターです。
- **<version>**は、メタデータを適用するファイルの特定のバージョンです。これはオプション・パラメーターです。
- **<label>**は、メタデータを適用するファイルのラベル付きバージョンです。これはオプション・パラメーターです。

- `<topic>`は、ファイルまたはフォルダーに適用するトピックです。これはオプション・パラメーターです。
- `<customProperty>`は、ファイルまたはフォルダーに適用するカスタム・プロパティ値です。これはオプション・パラメーターです。この値は`<customProperty>=<value>`という形式で指定します。複数のカスタム・プロパティを適用するには、区切り文字としてセミコロン(;)を使用します(`<customProperty>=<value>;<customProperty>=<value>`)。複数選択のプロパティ値は、|演算子(`<customProperty>=opt1|opt2;<customProperty>=value`)で区切ります。

注: setMetadata API を使用する場合は、1 つ以上のオプション・パラメーターを指定する必要があります。

例

```
python CADSTool.py setMetadata --source "/Demo/Drafts/MyReport.rptdesign" --version
"0:2006-08-25T21:15:49+01:00" --keyword "Quarterly"
--customProperty "multi=hi|hello|bye;Complexity Degree=Simple" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- Metadata set successfully on `<source>`.
- Metadata set successfully on `<source>` with version `<version>`.
- `<source>` does not exist
- `<source>` Error setting metadata.

uploadFile 操作

ローカル・ファイル・システム上のファイルをコンテンツ・リポジトリに保存します。ファイルが既に存在する場合に新しいバージョンのファイルを作成するオプションがあります。

構文

```
python CADSTool.py uploadFile --source "<source>" --target "<target>" --createVersion -z
```

ここで、

- `<source>`は、アップロードするファイルの(ローカル・ファイル・システム上の)完全修飾パスです。これは必須パラメーターです。
- `<target>`は、ファイルのアップロード先となるリポジトリ内のフォルダーの完全修飾パスです。これは必須パラメーターです。
- `--createVersion` は、指定されたファイルが既に存在する場合に、新しいバージョンを作成することを指定します。これはオプション・パラメーターです。

例

以下の例では、アップロード先を *Drafts* の完全修飾パスにしています。

```
python CADSTool.py uploadFile --source "C:\Demo\MyReport.rptdesign"
--target "/Demo/Drafts" -z
```

/Demo/Drafts フォルダーに既に MyReport.rptdesign が存在する場合は、以下のように `--createVersion` パラメーターを指定します。

```
python CADSTool.py uploadFile --source "C:\Demo\MyReport.rptdesign"
--target "/Demo/Drafts" --createVersion -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> uploaded successfully to <target>.
- <source> does not exist.
- <target> does not exist.
- <source> Error Uploading File.

プロセス管理機能

プロセス管理スクリプトには、ジョブを処理する機能が組み込まれています。以下の機能を備えています。

- ジョブの実行
- ジョブ履歴の取得
- ジョブ詳細の取得

このセクションでは、プロセス管理機能用スクリプトを Python コマンド・ラインで使用方法について説明します。すべての API について、詳細な構文の情報、例、予測されるメッセージを記載しています。

キーワード

24 ページの表 5 に、プロセス管理 API でサポートされるキーワードを示します。2 列目には、オプションで使える短縮形のキーワードを示しています。この表には、プロセス管理 API 固有のキーワードだけを記載しています。プロセス管理 API とリポジトリ API の両方に適用されるその他のキーワードについては、3 ページの表 2 と 5 ページの表 3 を参照してください。

表 5. プロセス管理 API のキーワード		
キーワード	オプションの短縮形	使用法
--source	-s	ソース・ジョブ (パスを含む)
--target	-t	ターゲット・フォルダー・パス
--notification	-j	ジョブを通知ありで実行することを指定します。
-- async	-m	ジョブを非同期で実行することを指定します
--execId	-y	ジョブの実行 ID
--jobStepName	-q	ジョブ・ステップの名前
--log		ログを削除しないことを指定します。このキーワードを --target とともに指定した場合は、--target で指定した場所にログが格納されます。指定しない場合、ログはインラインで表示されます。

プロセス管理の操作

deleteJobExecutions 操作

指定されたジョブ実行オブジェクトを削除します。

構文

```
python CADSTool.py deleteJobExecutions --execIds "<execIDs>" -z
```

<execIDs>の値は、削除する実行の ID をスペースで区切ったリストです。これは必須パラメーターです。

例

```
python CADSTool.py deleteJobExecutions --execIds  
"0a58c33d002ce9080000 010e0ccf7b01800e" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- Execution Id not specified.
- Job executions deleted successfully. Execution Id is <execIDs>.

executeJob 操作

渡されたパラメーターに基づいて、ジョブを同期的または非同期的に実行します。同期実行の場合は、ジョブが完了するまで API は返されません。非同期実行の場合は、ジョブの開始後に API が返されます。

構文

```
python CADSTool.py executeJob --source "<source>" --notification -- async -z
```

ここで、

- <source>は、リポジトリ内のジョブの完全修飾パスです。これは必須パラメーターです。
- --notification は、通知付きでジョブを実行する場合に使用します。これはオプション・パラメーターです。
- -- async は、ジョブを非同期に実行するために使用されます。これはオプション・パラメーターです。

例

以下の例では、ジョブの実行を通知せず、ジョブを同期的に実行します。

```
python CADSTool.py executeJob --source "/Demo/Jobs/Reports" -z
```

以下の例では、ジョブの実行を通知し、ジョブを同期的に実行します。

```
python CADSTool.py executeJob --source "/Demo/Jobs/Reports" --notification -z
```

以下の例では、ジョブの実行を通知せず、ジョブを非同期的に実行します。

```
python CADSTool.py executeJob --source "/Demo/Jobs/Reports" -- async -z
```

以下の例では、ジョブの実行を通知し、ジョブを非同期的に実行します。

```
python CADSTool.py executeJob --source "/Demo/Jobs/Reports" -- async --notification -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- <source> Job executed successfully. Execution Id is <execId>.
- <source> No such job exists. Please try again.
- <source> Error executing job.

executeJobWithOptions 操作

渡されたパラメーターに基づいて、オプション付きのジョブを同期的または非同期的に実行します。同期実行の場合は、ジョブが完了するまで API は返されません。非同期実行の場合は、ジョブの開始後に API が返されます。

構文

```
python CADSTool.py executeJobWithOptions --source "source" --options "option1=value1;option2=value2" --notification --async -z
```

ここで、

- `--source` は、リポジトリ内のジョブの完全修飾パスです。これは必須パラメーターです。
- `--options` は、このジョブを実行するためのオプションです。これは必須パラメーターです。
- `--notification` は、通知付きでジョブを実行する場合に使用します。これはオプション・パラメーターです。
- `--async` は、ジョブを非同期に実行するために使用されます。これはオプション・パラメーターです。

例

以下の例では、ジョブの実行を通知せず、ジョブを同期的に実行します。

```
python CADSTool.py executeJobWithOptions --source "/Demo/Jobs/Reports" --options "command=dir" -z
```

以下の例では、ジョブの実行を通知し、ジョブを同期的に実行します。

```
python CADSTool.py executeJobWithOptions --source "/Demo/Jobs/Reports" --options "command=dir;path=C:" --notification -z
```

以下の例では、ジョブの実行を通知せず、ジョブを非同期的に実行します。

```
python CADSTool.py executeJobWithOptions --source "/Demo/Jobs/Reports" --options "command=dir;path=C:" --async -z
```

以下の例では、ジョブの実行を通知し、ジョブを非同期的に実行します。

```
python CADSTool.py executeJobWithOptions --source "/Demo/Jobs/Reports" --options "command=dir;path=C:" --async --notification -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- "source" Job executed successfully. Execution Id is <execId>.
- "source" No such job exists. Please try again.
- "source" Error executing job.

getJobExecutionDetails 操作

ジョブ・ステップや繰り返しなど、特定のジョブの実行の詳細をリストします。

構文

```
python CADSTool.py getJobExecutionDetails --execId "<execID>" --log --target  
"<target>" -z
```

ここで、

- `<execId>` は、ジョブの実行 ID です。これは必須パラメーターです。
- `--log` は、ジョブ・ログをインラインで表示することを指定します。 `--log` パラメーターを指定しなかった場合、ジョブ・ステップの実行によって生成されたログは表示されません。これはオプション・パラメーターです。
- `<target>` は、ログを保管する (ローカル・ファイル・システム上の) 場所です。これはオプション・パラメーターです。このパラメーターを指定する場合は、`--log` パラメーターも指定する必要があります。

例

以下の例では、特定のジョブ実行の詳細をリストします。

```
python CADSTool.py getJobExecutionDetails --execId "0a58c3710016a7860000010d1a6a87b48400" -z
```

以下の例では、特定のジョブ実行の詳細をリストし、ログをインラインで表示します。

```
python CADSTool.py getJobExecutionDetails --execId "0a58c3710016a7860000010d1a6a87b48400" --log -z
```

以下の例では、特定のジョブ実行の詳細をリストし、ジョブ・ログを特定の場所に保存します。

```
python CADSTool.py getJobExecutionDetails --execId "0a58c3710016a7860000010d1a6a87b48400" --log --target "c:\logs" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- ジョブが正常に実行された場合は、ジョブ、ジョブ・ステップ、ジョブの繰り返しに関する実行の詳細がすべてリストされます。ログは、インラインで表示されるか、ローカル・ファイル・システム上の特定の場所に保存されます。
- <execId> No such execution exists. Please try again.
- <execId> Error displaying details of a job execution.
- Target cannot be specified without log parameter

getJobExecutionList 操作

特定のジョブのすべてのバージョンについて、現在の実行と完了した実行をリストします。

構文

```
python CADSTool.py getJobExecutionList --source "<source>" -z
```

<source>の値は、リポジトリ内のジョブの完全修飾パスです。これは必須パラメーターです。

例

```
python CADSTool.py getJobExecutionList --source "/Demo/Jobs/Reports" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- 指定したジョブが正常に実行された場合は、実行 ID、ジョブ名、ジョブの実行状況、ジョブ実行の開始時刻と終了時刻など、実行に関する詳細がすべてリストされます。
- <source> does not exist.
- <source> Error displaying execution list for a job.

cancelJob 操作

実行中のジョブをキャンセルします。

構文

```
python CADSTool.py cancelJob --execId "execIDs" -z
```

"execIDs"の値は、キャンセルする実行の ID のリスト (区切りはスペース) です。これは必須パラメーターです。

例

```
python CADSTool.py cancelJob --execId "0a58c33d002ce9080000010e0ccf7b01800e" -z
```

メッセージ

この API を使用すると、以下のメッセージが表示されることがあります。

- Execution Id not specified.

第 3 章 PESImpl モジュール

IBM スポス コラボレーションおよびデプロイメント・サービス・パイソンのためのエッセンシャルの機能により、Python スクリプト内から直接 IBM スポス コラボレーションおよびデプロイメント・サービス・リポジトリ オブジェクトと対話することができます。

Python コード内で、`pes.api.PESImpl` モジュールから `PESImpl` クラスをインポートします。次に、接続先リポジトリの接続情報を使用する `PESImpl` オブジェクトを作成します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl("user", "password", "host", "port", ssl=True, "appserver_type")
```

`PESImpl` コンストラクターのパラメーターは以下のとおりです。

- `user` は、ユーザーの名前に対応します。ネイティブ・プロバイダーからのユーザーではない場合は、値にセキュリティ・プロバイダーを表すプレフィックスが含まれていなければなりません。プレフィックスとしては以下の値が有効です。
 - `Native` (システム固有のネイティブ・ローカル・セキュリティ・プロバイダーの場合)。これはデフォルト・プロバイダーです。
 - `AD_<name>` for Active Directory。ここで、`<name>` は、システム内のセキュリティ・プロバイダー名に対応します。
 - `ADL_<name>` for Active Directory (ローカル・オーバーライド)。ここで、`<name>` は、システム内のセキュリティ・プロバイダー名に対応します。
 - `ldap_<name>` (OpenLDAP の場合)。ここで、`<name>` は、システム内のセキュリティ・プロバイダー名に対応します。

プレフィックスの後ろにスラッシュを付け、その後にユーザー名を続けます。Active Directory プロバイダーの場合は、プレフィックスにドメインを含めます。例えば、Active Directory インスタンス `AD_SPSSAD` の `MYDOMAIN` ドメイン内のユーザー `icrod` の場合、値は `AD_SPSSAD/MYDOMAIN/icrod` となります。ユーザー `icrod` が OpenLDAP プロバイダー `SPSSLDAP` 内にある場合は、値は `ldap_SPSSLDAP/icrod` となります。

- `password` は、指定したユーザーに関連付けられている暗号化されたパスワードに対応します。
- `host` は、リポジトリ・サーバーの名前を指定します。
- `port` は、リポジトリ・サーバーのポート番号を指定します。
- `ssl=True` は、リポジトリ・サーバーが Secure Sockets Layer (SSL) プロトコルを使用して通信を暗号化することを指定します。`PESImpl` オブジェクトの作成時に `ssl` パラメーターを `False` に設定した場合、またはこのパラメーターを指定しなかった場合は、サーバー通信で SSL が使用されません。SSL を使用する場合は、リポジトリ・サーバーを SSL 用に構成する必要があります。詳しくは、管理者用の資料を参照してください。

`host` パラメーター、`port` パラメーター、`ssl` パラメーターを指定する代わりに、サーバーの URL を指定することもできます。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl("myUser", "myPass", appserver_type="appserver_type" server_url="url")
```

`server_url` パラメーターは、リポジトリ・サーバーの完全な URL を指定します。サーバーでカスタムのコンテキスト・パスを使用する場合は、このパラメーターを使用してください。例えば以下のコンストラクターは、ポート 443 で SSL を使用し、コンテキスト・パスが `/ibm/spss` である、`myserver` というサーバーに対応します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl("myUser", "encryptedPass", "websphere" server_url="https://myserver:443/ibm/spss")
```

注: IPv6 アドレスは、`[3ffe:2a00:100:7031::1]` のように大括弧で囲む必要があります。

`pesImpl` オブジェクトを使用して、特定のメソッドにアクセスすることができます。

コンテンツ・リポジトリ API

コンテンツ・リポジトリ・スクリプトには、ファイルやフォルダーなどのリポジトリ・リソースを処理する機能が組み込まれています。以下の機能を備えています。

- フォルダーの作成および削除
- ファイルのアップロードおよびダウンロード
- フォルダーのエクスポートおよびインポート
- ラベル、セキュリティ、およびメタデータの管理

このセクションでは、リポジトリに格納されているリソースの処理で使用する PESImpl の API の概要について説明します。すべてのメソッドについて、詳細な構文の情報、例、予測されるメッセージを記載しています。

メソッド

以下の各セクションでは、IBM スポス コラボレーションおよびデプロイメント・サービス でサポートされるすべてのコンテンツ・リポジトリ・メソッドを示します。

注:

- オプション・パラメーター **Label** および **Version** を持つすべてのメソッドに対して、**Label** または **Version** のいずれかを使用しますが、両方は使用しません。対象のファイルまたはフォルダーの **Version** と **Label** を指定しなかった場合は、最新のバージョンが使用されます。
- リポジトリ内のファイルまたはフォルダーのパスが必要なメソッドでは、パスとオブジェクト URI のいずれか一方を使用することができます。オブジェクト URI を調べるには、IBM SPSS デプロイメント・マネージャー でオブジェクトのプロパティを表示します。
- ラテン文字以外の Unicode 文字を含むソース・リポジトリ、ターゲット・リポジトリ、またはファイル・システムのパスを入力する必要があるメソッドの場合は、文字列を Unicode オブジェクトとして指定する必要があります。以下に例を示します。

```
identificationSpecifier = pesImpl.uploadFile  
(source=u'C:\Analytics\La Peña.txt',  
 target=u'/La Peña')
```

advanceSearch メソッド

入力として渡されたさまざまなパラメーターに基づき、リポジトリ内でファイルやフォルダーを検索します。

以下の項目を検索できます。

- 作成者
- 説明
- 役職
- 作成者
- 変更者
- 有効期間の開始日
- 有効期間の終了日
- MIME タイプ
- ラベル
- キーワード
- トピック
- 作成開始日
- 作成終了日

- バージョンの変更開始日
- オブジェクトの変更終了日
- オブジェクトの変更開始日
- バージョンの変更開始日
- バージョンの変更終了日
- 親フォルダー URI
- リソース URI

```
advanceSearch(criteriaDict,submittedHierarchy)
```

表 6. *advanceSearch* の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
<i>criteriaDict</i>	必須	辞書	この辞書には、検索基準となるキーと値のペアが格納されます。使用可能なキー値は次のとおりです。 • author • title • description • createdBy • modifiedBy • expirationStartDate • expirationEndDate • mimeType • label • keyword • topic • createdStartDate • createdEndDate • objectModifiedStartDate • objectModifiedEndDate • versionModifiedStartDate • versionModifiedEndDate • parentURI • uri	<pre>{ "author": "admin" , "title": "search" , "label": "label1" }</pre>

表 6. <i>advanceSearch</i> の入力パラメーター (続き)				
フィールド	用途	タイプ	説明	サンプルの値
<i>submittedHierarchy</i>	オプション	ブール値	「 <i>Submitted Jobs</i> 」フォルダーを検索するかどうかを指定します	True または False

現時点では、*expirationStartDate* と *expirationEndDate* を他の検索フィールド (*title* や *author* など) と併用することはできません。

表 7. <i>advanceSearch</i> の戻り値	
タイプ	説明
<i>PageResult</i>	各行が、検索で一致した結果に対応している構造体。詳しくは、 62 ページの『PageResult クラス』 のトピックを参照してください。

表 8. <i>advanceSearch</i> の例外	
タイプ	説明
<i>InsufficientParameterException</i>	必須パラメーターが指定されていません。

例: ラベルおよびキーワードによる検索

以下の例では、*Production* というラベルが付いていて、*keyword* の値が *Quarterly* となっているすべてのファイル・バージョンを返します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
critDict = {'label': 'Production', 'keyword': 'Quarterly'}
sResults = pesImpl.advanceSearch(critDict)
sRows = sResults.getRows()
for sRow in sRows:
    print("Author: ", sRow.getAuthor())
    print("Title: ", sRow.getTitle())
    for child in sRow.getChildRow():
        print("Version: ", child.getVersionMarker())
        print("Label: ", child.getVersionLabel())
        print("Keywords: ", child.getKeyword())
        print("URI: ", child.getUri())
```

例: URI による検索

以下の例では、指定された URI を持つファイルのすべてのバージョンを返します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
critDict = {'uri': 'spsscr:///id=a010a37ba5992bb00000127b0f952f945be'}
sResults = pesImpl.advanceSearch(critDict)
sRows = sResults.getRows()
for sRow in sRows:
    print("Author: ", sRow.getAuthor())
    print("Title: ", sRow.getTitle())
    for child in sRow.getChildRow():
        print("Version: ", child.getVersionMarker())
        print("Label: ", child.getVersionLabel())
        print("Keywords: ", child.getKeyword())
        print("URI: ", child.getUri())
```

applySecurity メソッド

リポジトリ内のファイルまたはフォルダーに対するセキュリティー・アクセス・コントロール・リスト (ACL) を設定します。

```
applySecurity(source, principal, permission, provider, cascade)
```

表 9. <i>applySecurity</i> の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
<i>source</i>	必須	ストリング	リポジトリ内のファイルまたはフォルダーの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/ Temp.txt" または "0a58c3670016a7860000 010dcee0eaa28219"
<i>principal</i>	必須	ストリング	ACL の一部として指定のファイルまたはフォルダーに適用するユーザー (<i>admin</i> など)。	admin
<i>permission</i>	必須	ストリング	指定したファイルまたはフォルダーに適用するアクセス権のタイプ。	READ, WRITE, DELETE, MODIFY_ACL, OR OWNER
<i>provider</i>	オプション	ストリング	セキュリティをユーザーに適用するために使用するセキュリティ・プロバイダー (<i>Native</i> など)。有効な値は次のとおりです。 • <i>Native</i> (システム固有のネイティブ・ローカル・セキュリティ・プロバイダーの場合)。これはデフォルト・プロバイダーです。 • <i>AD_<name></i> for Active Directory。ここで、<name> は、システム内のセキュリティ・プロバイダー名に対応します。 • <i>ADL_<name></i> for Active Directory (ローカル・オーバーライド)。ここで、<name> は、システム内のセキュリティ・プロバイダー名に対応します。 • <i>ldap_<name></i> (OpenLDAP の場合)。ここで、<name> は、システム内のセキュリティ・プロバイダー名に対応します。	Native
<i>cascade</i>	オプション	ブール値	セキュリティ設定を、指定したフォルダー内のすべてのファイルとサブフォルダーに適用します。	True または False

表 10. <i>applySecurity</i> の戻り値	
タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 11. <i>applySecurity</i> の例外	
タイプ	説明
<code>ResourceNotFoundException</code>	ソース・ファイルが存在しません。
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。
<code>IllegalParameterException</code>	指定したユーザーまたはセキュリティー・プロバイダーの名前が正しくありません。

例

以下の例では、指定のファイルに対する *READ* 権限をユーザーに割り当てます。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
bSuccess = pesImpl.applySecurity(source="/Projects",principal="icrod",permission="READ", provider="Native")
```

cascadeSecurity メソッド

フォルダーのセキュリティー設定を、そのフォルダー内のすべてのファイルとサブフォルダーに適用します。

```
cascadeSecurity(source)
```

表 12. <i>cascadeSecurity</i> の入力パラメーター				
フィールド	使用	タイプ	説明	サンプルの値
<i>source</i>	必須	ストリング	リポジトリ内のフォルダーの完全修飾パスまたはオブジェクト URI。	<code>"/Temp Folder"</code> または <code>"0a58c3670016a7860000010dcee0eaa28219"</code>

表 13. <i>cascadeSecurity</i> の戻り値	
タイプ	説明
<code>Boolean</code>	<code>True</code> または <code>False</code> 。メソッドが正常に実行されたかどうかを示します。

表 14. <i>cascadeSecurity</i> の例外	
タイプ	説明
<code>ResourceNotFoundException</code>	ソース・フォルダーが存在しません。
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。

例

以下の例では、*Projects* フォルダーのセキュリティーをそのフォルダーの内容に連鎖的に適用します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
bSuccess = pesImpl.cascadeSecurity(source="/Projects")
```

copyResource メソッド

ファイルまたはフォルダーをリポジトリ内の別のフォルダーにコピーします。指定したソース・ファイルまたはフォルダーをコピーするときに、名前を変更できます。

名前の変更について詳しくは、[53 ページの『moveResource メソッド』](#)を参照してください。

```
copyResource(source, target)
```

表 15. copyResource の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	リポジトリ内のファイルまたはフォルダーの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/Temp.txt" または "0a58c3670016a7860000010dcee0eaa28219"
ターゲット	必須	ストリング	ファイルのコピー先フォルダーの完全修飾パスまたはオブジェクト URI。新しいファイル名を指定して、指定したファイルまたはフォルダーをコピーするときに名前を変更することもできます。	"/New Folder" または "/New Folder/abc.dat"

表 16. copyResource の戻り値

タイプ	説明
String	コピーされるファイルまたはフォルダーの URI。

表 17. copyResource の例外

タイプ	説明
ResourceNotFoundException	ソース・ファイルまたはターゲット・フォルダーが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。

例

以下の例では、*Drafts* フォルダーを *Projects* というフォルダーにコピーします。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KLl0Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
uri = pesImpl.copyResource(source="/Demo/Drafts/MyReport.rptdesign",target="/Projects") print(uri)
```

createFolder メソッド

リポジトリ内の指定した場所に新しいフォルダーを作成します。

```
createFolder(source)
```


表 18. createFolder の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	リポジトリに作成するフォルダー	/Temp Folder/Sample Folder

表 19. createFolder の戻り値	
タイプ	説明
String	作成されたフォルダーの URI。

表 20. createFolder の例外	
タイプ	説明
InsufficientParameterException	必須パラメーターが指定されていません。
ResourceAlreadyExistsException	指定したフォルダーはリポジトリ内に既に存在します。

例

以下の例では、*Demo* フォルダーの子として *Drafts* というフォルダーを作成します。フォルダーの作成で問題が発生した場合は、例外メッセージがコンソールに送信されます。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ=]", host = "localhost", port = "8080",
appserver_type = "websphere")
try:
    uri = pesImpl.createFolder(source="/Demo/Drafts")
    print("URI for the folder is:", uri)
except:
    print("Unhandled exception in createFolder.")
```

deleteFile メソッド

ファイルをリポジトリから削除します。ファイルのすべてのバージョンを削除します。

```
deleteFile(source,submittedHierarchy)
```

表 21. deleteFile の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	リポジトリ内のファイルの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/Temp.txt" または "0a58c3670016a7860000010dcee0eaa28219"
submittedHierarchy	オプション	ブール値	ファイルが「Submitted Jobs」フォルダーに存在するかどうかを示します。	True または False

表 22. <code>deleteFile</code> の戻り値	
タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 23. <code>deleteFile</code> の例外	
タイプ	説明
<code>ResourceNotFoundException</code>	ソース・ファイルが存在しません。
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。
<code>IllegalParameterException</code>	削除対象として指定されたリソースはフォルダーです。

例

以下の例では、`MyReport.rptdesign` ファイルをリポジトリから削除します。

```
from pes.util.PESExceptions import *
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KLl0Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
try:
    bSuccess = pesImpl.deleteFile(source="/Demo/Drafts/MyReport.rptdesign")
except ResourceNotFoundException as rnf:
    print("Specified file does not exist.")
except InsufficientParameterException as ipe:
    print("No file specified.")
except IllegalParameterException as ipe:
    print("Item to be deleted is not a file.")
```

```
from pes.util.PESExceptions import * from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KLl0Yo406545tgGsYQ==]", host =
"localhost", port = "8080", appserver_type = "websphere")
try:
    bSuccess = pesImpl.deleteFile(source="/Demo/Drafts/MyReport.rptdesign")
except ResourceNotFoundException as rnf:
    print("Specified file does not exist.")
except InsufficientParameterException as ipe:
    print("No file specified.")
except IllegalParameterException as ipe:
    print("Item to be deleted is not a file.")
```

`deleteFileVersion` メソッド

リポジトリから、ファイルの特定のバージョンを削除します。

```
deleteFileVersion(source,version,label,submittedHierarchy)
```

表 24. <code>deleteFileVersion</code> の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
<code>source</code>	必須	スト リ ン グ	リポジトリ内のファイルの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/ Temp.txt" または "0a58c3670016a78600 00010dcee0eaa28219"

表 24. <i>deleteFileVersion</i> の入力パラメーター (続き)				
フィールド	用途	タイプ	説明	サンプルの値
バージョン	オプション。ただし、 <i>version</i> または <i>label</i> のいずれかを必ず指定する必要があります。	ストリング	削除するファイルの具体的なバージョン。	"0:2006-08-25 21:15:49.453"
ラベル	オプション。ただし、 <i>version</i> または <i>label</i> のいずれかを必ず指定する必要があります。	ストリング	削除するファイルの具体的なラベル付きバージョン。	"Version 1"
<i>submittedHierarchy</i>	オプション	ブール値	ファイルが「Submitted Jobs」フォルダーに存在するかどうかを示します。	True または False

表 25. <i>deleteFileVersion</i> の戻り値	
タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 26. <i>deleteFileVersion</i> の例外	
タイプ	説明
ResourceNotFoundException	ソース・ファイルまたはターゲット・フォルダーが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。
IllegalParameterException	削除対象として指定されたリソースはフォルダーです。

例

以下の例では、*Test* というラベルが付いた *MyReport.rptdesign* ファイルのバージョンをリポジトリから削除します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ=]", host = "localhost", port = "8080",
appserver_type = "websphere")
bSuccess = pesImpl.deleteFileVersion(source="/Demo/Drafts/MyReport.rptdesign",label="Test")
```

deleteFolder メソッド

フォルダーとその内容をリポジトリから削除します。

```
deleteFolder(source,submittedHierarchy)
```

表 27. deleteFolder の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	リポジトリ内のフォルダーの完全修飾パスまたはオブジェクト URI。	"/Temp Folder" または "0a58c3670016a7860000010dcee0eaa28219"
submittedHierarchy	オプション	ブール値	フォルダーが「送信済みジョブ」フォルダーにあるかどうかを示します。	True または False

表 28. deleteFolder の戻り値

タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 29. deleteFolder の例外

タイプ	説明
ResourceNotFoundException	指定したフォルダーは存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。
IllegalParameterException	削除対象として指定されたリソースはフォルダーではありません。

例

以下の例では、*Drafts* という名前のフォルダーをリポジトリから削除します。フォルダーの削除で問題が発生した場合は、例外メッセージがコンソールに送信されます。

```
from pes.util.PESExceptions import *
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KLl0Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
try:
    bSuccess = pesImpl.deleteFolder(source="/Demo/Drafts")
except ResourceNotFoundException as rfe:
    print ("Specified folder does not exist.")
except InsufficientParameterException as ipe:
    print ("No folder specified.")
except IllegalParameterException as ipe:
    print ("Item to be deleted is not a folder.")
```

downloadFile メソッド

リポジトリ内の特定のバージョンのファイルをローカル・ファイル・システムにダウンロードします。

```
downloadFile(source,target,version,label)
```

表 30. downloadFile の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	ダウンロードするファイルの完全修飾リポジトリ・パス。	"/Temp Folder/Temp.txt"

表 30. `downloadFile` の入力パラメーター (続き)

フィールド	用途	タイプ	説明	サンプルの値
ターゲット	必須	ストリング	ファイルのダウンロード先フォルダーの (ローカル・ファイル・システム上の) 完全修飾パス。	"C:\Temp"
バージョン	オプション。バージョンまたはラベルのいずれかを指定できます。	ストリング	ダウンロードするファイルの具体的なバージョン。	"0:2006-08-25 21:15:49.453"
ラベル	オプション。バージョンまたはラベルのいずれかを指定できます。	ストリング	ダウンロードするファイルの具体的なラベル付きバージョン。	"Version 2"

表 31. `downloadFile` の戻り値

タイプ	説明
Resource	リポジトリ・オブジェクトに関する情報のコンテナ。詳しくは、 60 ページの『Resource クラス』 のトピックを参照してください。

表 32. `downloadFile` の例外

タイプ	説明
ResourceNotFoundException	ソース・ファイルまたはターゲット・フォルダーが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。

例

以下の例では、`MyReport.rptdesign` ファイルの *Production* というラベルが付いたバージョンを、ローカル・ファイル・システムの *Shared* ディレクトリーにダウンロードします。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
  appserver_type = "websphere")
resource = pesImpl.downloadFile(source="/Demo/Drafts/MyReport.rptdesign",
  target="c:/Demo/Shared",label="Production")
```

exportResource メソッド

指定されたリポジトリ・フォルダーを、ローカル・ファイル・システム上の指定された *.pes エクスポート・ファイルにエクスポートします。

```
exportResource(source, target)
```

表 33. *exportResource* の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
<i>source</i>	必須	ストリング	エクスポートするフォルダーの完全修飾リポジトリ・パスまたはオブジェクト URI。	/Temp Folder または 0a58c3670016a78 60000010dcee0eaa28219
ターゲット	必須	ストリング	フォルダーのエクスポート先となる (ローカル・ファイル・システム上の) 完全修飾パスとファイル名。	C:\Temp\backup.pes

表 34. *exportResource* の戻り値

タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 35. *exportResource* の例外

タイプ	説明
ResourceNotFoundException	ソース・ファイルまたはターゲット・フォルダーが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。
IllegalParameterException	指定されたターゲットはフォルダーです。ターゲットは *.pes ファイルでなければなりません。

例

以下の例では、*Drafts* フォルダーの内容を、ローカル・ファイル・システムの *backups* フォルダーのエクスポート・ファイルにエクスポートします。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ=]", host = "localhost", port = "8080",
appserver_type = "websphere")
bSuccess = pesImpl.exportResource(source="/Projects",target="C:\Demo\drafts.pes")
```

getAccessControllist メソッド

リポジトリ内の指定されたファイルまたはフォルダーのセキュリティー・アクセス・コントロール・リスト (ACL) を取得します。

```
getAccessControllist(source,submittedHierarchy)
```

表 36. `getAccessControllList` の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
<code>source</code>	必須	ストリング	リポジトリ内のファイルまたはフォルダーの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/ Temp.txt" または "0a58c3670016a78600 00010dcee0eaa28219"
<code>submittedHierarchy</code>	オプション	ブール値	ファイルまたはフォルダーが「Submitted Jobs」フォルダーに存在するかどうかを示します。	True または False

表 37. `getAccessControllList` の戻り値

タイプ	説明
Dictionary	ユーザー名とそれに関連する権限が格納された辞書が表示されます。以下に例を示します。 <pre>{ "admin": "MODIFY_ACL", "Joe": "DELETE" }</pre>

表 38. `getAccessControllList` の例外

タイプ	説明
<code>ResourceNotFoundException</code>	ソース・ファイルまたはターゲット・フォルダーが存在しません。
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。

例

以下の例では、`MyReport.rptdesign` ファイルの ACL を出力します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES]KrY+KL10Yo406545tgGsYQ==", host = "localhost", port = "8080",
appserver_type = "websphere")
aclDic = pesImpl.getAccessControllList(source = "/Projects/MyReport.rptdesign")
print (aclDic)
```

`getAllVersions` メソッド

リポジトリ内のファイルのすべてのバージョンのリストを取得します。

```
getAllVersions(source, submittedHierarchy)
```

表 39. <i>getAllVersions</i> の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
<i>source</i>	必須	ストリング	リポジトリ内のファイルの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/Temp.txt" または "0a58c3670016a7860000010dcee0eaa28219"
<i>submittedHierarchy</i>	オプション	ブール値	ファイルが「 <i>Submitted Jobs</i> 」フォルダーに存在するかどうかを示します。	True または False

表 40. <i>getAllVersions</i> の戻り値	
タイプ	説明
List	リソース・オブジェクトのリスト。 60 ページの『Resource クラス』 を参照してください。

表 41. <i>getAllVersions</i> の例外	
タイプ	説明
ResourceNotFoundException	ソース・ファイルが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。
IllegalParameterException	指定されたソースはフォルダーです。

例

この例では、*MyReport.rptdesign* ファイルのすべてのバージョンに関する情報を取得し、それぞれの作成者、バージョン・マーカ、バージョン・ラベルを出力します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES]KrY+KL10Yo406545tgGsYQ==", host = "localhost", port = "8080",
appserver_type = "websphere")
resourceList = pesImpl.getAllVersions(source="/Demo/Drafts/MyReport.rptdesign")
for resource in resourceList:
    print(resource['Resource']['Author']['value'])
    print(resource['Resource']['Version']['marker'])
    print(resource['Resource']['Version']['label'])
```

getChildren メソッド

指定されたりポジトリ・フォルダー内のすべてのファイルとフォルダーのリストを取得します。

```
getChildren(source,submittedHierarchy)
```


表 42. <i>getChildren</i> の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
<i>source</i>	必須	スト リ ン グ	リポジトリ内のフォルダーの完全修飾パスまたはオブジェクト URI。	"/Temp Folder" または "0a58c3670016a7860000010dcee0eaa28219"
<i>submittedHierarchy</i>	オプション	BOOLE AN	フォルダーが「送信済みジョブ」フォルダーにあるかどうかを示します。	True または False

表 43. <i>getChildren</i> の戻り値	
タイプ	説明
List	リソース・オブジェクトのリスト。60 ページの『 Resource クラス 』を参照してください。

表 44. <i>getChildren</i> の例外	
タイプ	説明
InsufficientParameterException	必須パラメーターが指定されていません。
ResourceNotFoundException	フォルダーが存在しません。

例

以下の例では、*/Demo/Drafts* フォルダーの内容を取得し、それぞれのタイトル、作成者、リソース ID を出力します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
resourceList = pesImpl.getChildren(source="/Demo/Drafts")
for resource in resourceList:
    print(resource['Resource']['Author']['value'])
    print(resource['Resource']['Version']['marker'])
    print(resource['Resource']['Version']['label'])
```

getCustomPropertyValue メソッド

指定されたカスタム・プロパティで利用できる有効な値を取得します。

```
getCustomPropertyValue(propertyName)
```

表 45. <i>getCustomPropertyValue</i> の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
<i>propertyName</i>	必須	スト リ ン グ	カスタム・プロパティの名前。	"FreeForm"

表 46. `getCustomPropertyValue` の戻り値

タイプ	説明
List	カスタム・プロパティで利用できる有効な値のリストを返します。プロパティの値を選択する必要がある場合 (単一選択や複数選択など)、このリストには選択可能な値がすべて格納されます。自由形式のプロパティの場合、このリストには、そのプロパティで利用できるデータ型 (String、Date、Number など) が格納されます。

表 47. `getCustomPropertyValue` の例外

タイプ	説明
ResourceNotFoundException	指定したプロパティは存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。

例

以下の例では、カスタム・プロパティ `Language` の値にアクセスします。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ=]", host = "localhost", port = "8080",
appserver_type = "websphere")
langList = pesImpl.getCustomPropertyValue(propertyName = "Language")
print (langList)
```

getMetadata メソッド

カスタム・プロパティやトピックの情報など、リポジトリ内のファイルやフォルダーのメタデータ属性を取得します。

```
getMetadata(source, version, label, submittedHierarchy)
```

表 48. `getMetadata` の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
<code>source</code>	必須	ストリング	リポジトリ内のファイルまたはフォルダーの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/ Temp.txt" または "0a58c3670016a7860 0 00010dcee0eaa28219 "
バージョン	オプション。バージョンまたはラベルのいずれかを指定できます。	ストリング	ファイルまたはフォルダーの具体的なバージョン。	"0:2006-08-25 21:15:49.453"

表 48. <i>getMetadata</i> の入力パラメーター (続き)				
フィールド	用途	タイプ	説明	サンプルの値
ラベル	オプション。バージョンまたはラベルのいずれかを指定できます。	ストリング	ファイルまたはフォルダーの具体的なラベル付きバージョン。	"Version 1"
<i>submittedHierarchy</i>	オプション	ブール値	ファイルが「 <i>Submitted Jobs</i> 」フォルダーに存在するかどうかを示します。	True または False

表 49. <i>getMetadata</i> の戻り値	
タイプ	説明
Resource	リポジトリ・オブジェクトに関する情報のコンテナ。詳しくは、 60 ページの『Resource クラス』 のトピックを参照してください。

表 50. <i>getMetadata</i> の例外	
タイプ	説明
ResourceNotFoundException	ソース・ファイルまたはフォルダーが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。

例

以下の例では、*/Demo/Drafts* フォルダーのリソース ID にアクセスします。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
resource = pesImpl.getMetadata(source="/Demo/Drafts")
resourceid = resource['ResourceID']['value']
```

importResource メソッド

ローカル・ファイル・システム上の既存の *.pes エクスポート・ファイルをリポジトリにインポートします。

```
importResource(source, target, resourceType, resourceConflict, invalidVersionConflict,
resourceDef, labelFrom, lockResolution, exclude)
```

表 51. <i>importResource</i> の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
<i>source</i>	必須	ストリング	インポートするファイルの (ローカル・ファイル・システム上の) 完全修飾パス。	"C:\Temp\New.pes"

表 51. <i>importResource</i> の入力パラメーター (続き)				
フィールド	用途	タイプ	説明	サンプルの値
ターゲット	必須	ストリング	インポート先フォルダーのリポジトリの完全修飾パスまたはオブジェクト URI。	"/Temp Folder" または "0a58c3670016a786000010dcee0eaa28219"
<i>resourceType</i>	オプション	ストリング	インポート対象のコンテンツのタイプ。次のいずれかの値を指定します。 • ContentRepository: ファイルやフォルダーなどのコンテンツ・オブジェクトの場合 • ResourceDef: リソース定義の場合 • Credential: ユーザー資格情報の場合 • DataSource: データ・ソース定義の場合 • MessageDomain: メッセージ・ドメインの場合 • ServerCluster: サーバー・クラスター定義の場合 • Server: サーバー定義の場合 • PromotionPolicy: プロモーション・ポリシーの場合 このパラメーターを指定しない場合は、デフォルト値の ContentRepository が使用されます。	

表 51. `importResource` の入力パラメーター (続き)

フィールド	用途	タイプ	説明	サンプルの値
<code>resourceConflict</code>	オプション	ストリング	重複した ID または名前の競合の解決方法を示します。次のいずれかの値を指定します。 • keepTarget. ターゲット項目が保持されます。 <code>.pes</code> ファイルに含まれる、ID が重複しているソース項目は、無視されます。 • addNewVersion. 通常、このオプションを使用すると ID または名前の競合が解決されます。ソース・オブジェクトとターゲット・オブジェクトの間で重複した ID の競合が発生した場合、新しいバージョンのオブジェクトが対象の場所に作成されます。名前の競合が発生した場合、対象の場所にあるインポートされたオブジェクトの名前が変更されます。通常、名前の変更が行われたオブジェクトには、<code>_1</code>、<code>_2</code>、などが付加されます。2つのバージョンのオブジェクトに同じラベルが付いている場合、2つのバージョンの同じ項目に同じラベルを使用することはできないため、システムは一方のラベルを保持し、重複したラベルを破棄します。保持されるラベルは、labelFrom パラメーターによって異なります。 このパラメーターを指定しない場合は、デフォルト値の keepTarget が使用されます。	
<code>labelFrom</code>	オプション	ストリング	オブジェクトの 2つのバージョンが同じラベルを持っている場合に使用するラベル。他方のバージョンのラベルは破棄されます。 source または target のいずれかを指定します。このパラメーターを指定しない場合は、デフォルト値の source が使用されます。	

表 51. `importResource` の入力パラメーター (続き)

フィールド	用途	タイプ	説明	サンプルの値
<code>lockResolution</code>	オプション	ストリング	ロックされたリソースが検出された場合の処理方法を定義します。次のいずれかの値を指定します。 • continue ロックされたリソースを省略して、インポートを続行します。 • abort ロックされたリソースが検出されると、インポート・プロセスを終了します。競合がオブジェクトのロックにより発生した場合は、インポート処理は終了され、失敗します。 このパラメーターを指定しない場合は、デフォルト値の abort が使用されます。	
<code>invalidVersionConflict</code>	オプション	ストリング	インポート処理中に無効なバージョンが検出された場合の処理方法を定義します。次のいずれかの値を指定します。 • import 無効なバージョンがインポートされます。 • discard 無効なバージョンは削除されます。 このパラメーターを指定しない場合は、デフォルト値の import が使用されます。	

表 51. `importResource` の入力パラメーター (続き)

フィールド	用途	タイプ	説明	サンプルの値
<code>resourceDef</code>	オプション	ストリング	リソース定義の処理動作を定義します。次のいずれかの値を指定します。 • recommended. リソース定義は、識別子または名前が対象の定義と競合しない場合にのみインポートされます。競合しているリソース定義はいずれもインポートされません。 • include. インポート・ファイル内のすべてのリソース定義がインポートされます。対応するチェック・ボックスを選択して、インポートから除外する 1 つ以上のリソース定義の種類を選択することができます。 • exclude. インポート・ファイルからのリソース定義はインポートされません。インポートされたオブジェクトは、使用可能なリソース定義を参照するように変更する必要が生じることがあります。 このパラメーターを指定しない場合は、デフォルト値の recommended が使用されます。	

表 51. `importResource` の入力パラメーター (続き)

フィールド	用途	タイプ	説明	サンプルの値
<code>EXCLUDE</code>	オプション	ストリング	インポート中に除外するリソース・タイプを定義します。複数の値を任意の順にセミコロンで区切って結合できます。以下の値を 1 つ以上指定します。 • credential: ユーザー資格情報を除外します • customproperty: リソース・オブジェクトのカスタム・プロパティを除外します • datasource: データ・ソース定義を除外します • messagedomain: メッセージ・ドメインを除外します • notification: 通知定義を除外します • servercluster: サーバー・クラスター定義を除外します • server: サーバー定義を除外します • topic: トピックの定義を除外します このパラメーターを指定しない場合は、すべてのタイプがインポートに含まれます。	

表 52. `importResource` の戻り値

タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 53. `importResource` の例外

タイプ	説明
<code>ResourceNotFoundException</code>	ソース・ファイルまたはターゲット・フォルダーが存在しません。
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。

例

以下のサンプルは、`drafts.pes` エクスポート・ファイルの内容を IBM スポス コラボレーションおよびデプロイメント・サービス・リポジトリの `/Demo/Drafts` フォルダーにインポートします。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
  appserver_type = "websphere")
bSuccess = pesImpl.importResource(source="C:\\Demo\\drafts.pes", target="/Demo/Drafts")
```


moveResource メソッド

ファイルまたはフォルダーをリポジトリ内の別のフォルダーに移動します。指定したソース・ファイルは、移動する際にファイル名を変更することができます。ターゲットの種類とターゲットの有無に応じて、最終的な名前が決まります。

ファイル移動時の名前変更機能の動作について、以下の表で説明します。

表 54. ファイル名変更		
ターゲット・タイプ	ターゲット・フォルダーが存在する	ターゲット・フォルダーが存在しない
フォルダー	ソース・ファイルがターゲット・フォルダーの子となります。	指定したターゲット・フォルダーの親フォルダーにソース・ファイルが移動され、ファイル名がターゲット・フォルダーの名前に変更されます。
ファイル	ターゲット・ファイルが存在するフォルダーにソース・ファイルが移動され、ファイル名がターゲットの名前に変更されます。	エラーが報告されます。

例えば、ソースがファイル `/Temp Folder/Temp.txt` で、指定されたターゲットがフォルダー `/Demo Folder` の場合、以下ようになります。

- フォルダー `Demo Folder` が存在する場合は、`Temp.txt` が `Demo Folder` に移動されます。
- フォルダー `Demo Folder` が存在しない場合、`Temp.txt` は `/` に移動され、`Demo Folder` に名前変更されません。

一方、ソースが `/Temp Folder/Temp.txt` で、指定されたターゲットがファイル `/Demo Folder/Abc.dat` の場合、以下ようになります。

- フォルダー `Demo Folder` が存在する場合は、`Temp.txt` が `Demo Folder` に移動され、名前が `Abc.dat` に変更されます。
- フォルダー `Demo Folder` が存在しない場合は、エラーが表示されます。

```
moveResource(source, target)
```

表 55. moveResource の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	リポジトリ内のファイルまたはフォルダーの完全修飾パスまたはオブジェクト URI。	<code>/Temp Folder/Temp.txt</code> または <code>0a58c3670016a7860000010dcee0eaa28219</code>
ターゲット	必須	ストリング	ファイル移動先フォルダーの完全修飾パスまたはオブジェクト URI。新しいファイル名を指定して、指定したファイルまたはフォルダーを移動するときに名前を変更することもできます。	<code>/New Folder</code> または <code>/New Folder/abc.dat</code>

表 56. moveResource の戻り値	
タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 57. moveResource の例外	
タイプ	説明
ResourceNotFoundException	指定したソースが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。

例

以下の例では、MyReport.rptdesign ファイルを /Demo/Drafts フォルダーから /Approved フォルダーに移動します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ=]", host = "localhost", port = "8080",
appserver_type = "websphere")
bSuccess = pesImpl.moveResource(source="/Demo/Drafts/MyReport.rptdesign",target="/Approved")
print (bSuccess)
```

removeLabel メソッド

リポジトリ内のファイルからラベルを削除します。

```
removeLabel(source, label)
```

表 58. removeLabel の入力パラメーター				
フィールド	用途	タイプ	サンプルの値	説明
source	必須	ストリング	"/Temp Folder/Temp.txt" または "0a58c3670016a7860000010dcce0eaa28219"	リポジトリ内のファイルの完全修飾パスまたはオブジェクト URI。
ラベル	必須	ストリング	"Version 1"	削除するラベル名。

表 59. removeLabel の戻り値	
タイプ	説明
String	更新されたファイルの URI。

表 60. removeLabel の例外	
タイプ	説明
ResourceNotFoundException	ソース・ファイルが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。

例

以下の例では、*MyReport.rptdesign* ファイルからラベル *Draft* を削除します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
uri = pesImpl.removeLabel(source="/Demo/Drafts/MyReport.rptdesign", label="Draft")
```

removeSecurity メソッド

```
removeSecurity(source,principal,provider,cascade)
```

リポジトリ内の指定されたファイルまたはフォルダーからセキュリティー・アクセス・コントロール・リスト (ACL) を削除します。

表 61. *removeSecurity* の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
<i>source</i>	必須	ストリング	リポジトリ内のファイルまたはフォルダーの完全修飾パスまたはオブジェクト URI。	<code>"/Temp Folder/Temp.txt"</code> または <code>"0a58c3670016a7860000010dcee0eaa28219"</code>
<i>principal</i>	必須	ストリング	指定のファイルまたはフォルダーから削除するユーザー (<i>admin</i> など)。	<code>admin</code>
<i>provider</i>	オプション	ストリング	ユーザーに関する情報の取得に使用するセキュリティー・プロバイダー (<i>Native</i> など)。有効な値は以下のとおりです。 • <i>Native</i> (システム固有のネイティブ・ローカル・セキュリティー・プロバイダーの場合)。これはデフォルト・プロバイダーです。 • <i>AD_<name></i> for Active Directory。ここで、<name> は、システム内のセキュリティー・プロバイダー名に対応します。 • <i>ADL_<name></i> for Active Directory (ローカル・オーバーライド)。ここで、<name> は、システム内のセキュリティー・プロバイダー名に対応します。 • <i>ldap_<name></i> (OpenLDAP の場合)。ここで、<name> は、システム内のセキュリティー・プロバイダー名に対応します。	<code>Native</code>

表 61. <code>removeSecurity</code> の入力パラメーター (続き)				
フィールド	用途	タイプ	説明	サンプルの値
<code>cascade</code>	オプション	ブール値	セキュリティ設定を、指定したフォルダー内のすべてのファイルとサブフォルダーに適用します。	True または False

表 62. <code>removeSecurity</code> の戻り値	
タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

表 63. <code>removeSecurity</code> の例外	
タイプ	説明
<code>ResourceNotFoundException</code>	ソース・ファイルまたはターゲット・フォルダーが存在しません。
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。
<code>IllegalParameterException</code>	指定したユーザーまたはセキュリティ・プロバイダーの名前が正しくありません。

例

以下の例では、`MyReport.rptdesign` ファイルからプリンシパルの ACL を削除します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
bSuccess = pesImpl.removeSecurity(source="/Projects/MyReport.rptdesign",principal="icrod")
```

search メソッド

リポジトリ内のファイルを検索し、検索基準に一致するメタデータ・コンテンツを持つファイル・バージョンのリストを返します。

```
search(criteria)
```

表 64. <code>search</code> の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
<code>criteria</code>	必須	ストリング	ファイル・メタデータを検索するために使用する値。	"Age"

表 65. <code>search</code> の戻り値	
タイプ	説明
<code>PageResult</code>	各行が、検索で一致した結果に対応している構造体。詳しくは、 62 ページの『PageResult クラス』 のトピックを参照してください。

表 66. <code>search</code> の例外	
タイプ	説明
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。

例

以下の例では、いずれかのメタデータ・フィールドに *Quarterly* というテキストが含まれているファイル・バージョンを検索します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
sResults = pesImpl.search(criteria="Quarterly")
sRows = sResults.getRows()
for sRow in sRows:
    print("Author: ", sRow.getAuthor())
    print("Title: ", sRow.getTitle())
    for child in sRow.getChildRow():
        print("Version: ", child.getVersionMarker())
        print("Label: ", child.getVersionLabel())
        print("Keywords:", child.getKeyword())
        print("URI:", child.getUri())
```

setLabel メソッド

リポジトリ内の任意のバージョンのファイルにラベルを適用します。既にファイルにラベルが付いている場合は、元のラベルを新しいラベルで置き換えます。

```
setLabel(source,version, label)
```

表 67. setLabel の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	リポジトリ内のファイルの完全修飾パスまたはオブジェクト URI。	"/Temp Folder/Temp.txt" または "0a58c3670016a7860000 010dcee0eaa28219"
バージョン	必須	ストリング	ファイルの具体的なバージョン。	"0:2006-08-25 21:15:49.453"
ラベル	必須	ストリング	ファイルに適用するラベル。	"Version 1"

表 68. setLabel の戻り値

タイプ	説明
String	更新されたファイルの URI。

表 69. setLabel の例外

タイプ	説明
ResourceNotFoundException	ソース・ファイルまたはバージョンが存在しません。
InsufficientParameterException	必須パラメーターが指定されていません。

例

以下の例では、*MyReport.rptdesign* ファイルの 2 番目のバージョンに *Beta* というラベルを割り当てます。Resource オブジェクトの *getVersionMarker* メソッドは、ラベルを付けるバージョンのマーカールを返します。

```
from pes.api.PESImpl import PESImpl
```

```
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ=]", host = "localhost", port = "8080",
  appserver_type = "websphere")
betaVersion = pesImpl.getAllVersions(source="/Demo/Drafts/MyReport.rptdesign")[1]['Resource']['Version']['marker']
print("Marker for the beta version is:", betaVersion)
uri = pesImpl.setLabel(source="/Demo/Drafts/MyReport.rptdesign", version=betaVersion,label="Beta")
```

setMetadata メソッド

メタデータ・プロパティをリポジトリ内のファイルとフォルダーに適用します。

以下の表に、メタデータ・プロパティを示します。また、それらのプロパティをファイルとフォルダーに適用できるかどうかを示します。

表 70. リポジトリ・オブジェクトのプロパティ	
メタデータ・プロパティ	リソース・タイプ
作成者	ファイル
説明	ファイルまたはフォルダー
役職	ファイルまたはフォルダー
有効期限	ファイルまたはフォルダー
キーワード	ファイル
トピック	ファイル
カスタム・メタデータ	ファイルまたはフォルダー

```
setMetadata(source,version,label,props)
```

表 71. setMetadata の入力パラメーター				
フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	リポジトリ内のファイルまたはフォルダーの完全修飾パスまたはオブジェクトURI。	"/Temp Folder/Temp.txt" または "0a58c3670016a7860000010dcee0eaa28219"
バージョン	オプション。バージョンまたはラベルのいずれかを指定できます。	ストリング	ダウンロードするファイルの具体的なバージョン。	"0:2006-08-25 21:15:49.453"

表 71. <i>setMetadata</i> の入力パラメーター (続き)				
フィールド	用途	タイプ	説明	サンプルの値
ラベル	オプション。バージョンまたはラベルのいずれかを指定できます。	ストリング	特定のバージョンのラベル。	"Label 1"
<i>props</i>	必須	辞書	設定するすべてのメタデータを、メタデータ名をキーとして持つ辞書に格納します。「値の例」の列に示しているように、このパラメーターは <i>topic</i> の値としてリストを取り、 <i>customProperty</i> には辞書を取ります。他のメタデータについては文字列を取ります。	<pre>{ 'author': 'admin', 'title': 'newTitle', 'description': 'desc', 'topic': [a,b], 'customProperty': { 'language': 'hindi english', 'FreeForm': 'abcd' } }</pre>

表 72. <i>setMetadata</i> の戻り値	
タイプ	説明
String	メタデータが設定されたファイルまたはフォルダーの URI。

表 73. <i>setMetadata</i> の例外	
タイプ	説明
<i>InsufficientParameterException</i>	必須パラメーターが指定されていません。
<i>ResourceNotFoundException</i>	ソース・ファイルまたはフォルダーが存在しません。

例

以下の例では、*MyReport.rptdesign* ファイルの *Production* バージョンにキーワード *Quarterly* を割り当てます。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
  appserver_type = "websphere")
pDict = {'keyword': 'Quarterly'}
uri = pesImpl.setMetadata(source="/Demo/Drafts/MyReport.rptdesign", version=prodVersion, props=pDict)
print(uri)
```

uploadFile メソッド

ローカル・ファイル・システム上のファイルをリポジトリに保存します。ファイルが既に存在する場合に新しいバージョンのファイルを作成するオプションがあります。

```
uploadFile(source,target,versionFlag)
```

表 74. `uploadFile` の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
<code>source</code>	必須	ストリング	アップロードするファイルの (ローカル・ファイル・システム上の) 完全修飾パス。	"C:\Temp\Temp.txt"
ターゲット	必須	ストリング	ファイルのアップロード先となる、リポジトリ内の宛先フォルダーの完全修飾パス。	"/Temp Folder"
<code>versionFlag</code>	オプション	ブール値	指定されたファイルが既に存在する場合に、新しいバージョンのファイルを作成します。	True または False

表 75. `uploadFile` の戻り値

タイプ	説明
String	アップロードされたファイルの URI。

表 76. `uploadFile` の例外

タイプ	説明
<code>ResourceNotFoundException</code>	ソース・ファイルまたはターゲット・フォルダーが存在しません。
<code>ResourceAlreadyExistsException</code>	ソース・ファイルと同じ名前のファイルまたはフォルダーがターゲット・フォルダーに存在していますが、 <code>versionFlag</code> パラメーターが指定されていません。
<code>InsufficientParameterException</code>	必須パラメーターが指定されていません。

例

この例では、`MyReport.rptdesign` ファイルをリポジトリ内の `/Demo/Drafts` フォルダーにアップロードします。ファイルが既に存在する場合は、`versionFlag` パラメーターを使用してファイルの新しいバージョンがアップロードされます。

```
from pes.util.PESExceptions import *
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KLl0Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
try:
    uri = pesImpl.uploadFile(source="C:\Demo\MyReport.rptdesign",target="/Demo/Drafts")
    print("URI for the uploaded file is: ", uri)
except ResourceAlreadyExistsException as rae:
    uri = pesImpl.uploadFile(source="C:\Demo\MyReport.rptdesign",target="/Demo/Drafts",versionFlag=True)
    print("URI for the uploaded file is: ", uri)
```

ラッパー・クラス

PESImpl API には、コンテンツ・リポジトリのメソッドによって呼び出された Web サービスから返されるオブジェクトのラッパーとして機能するクラスが用意されています。これらのラッパー・クラスは、メソッドから返された情報を表示するためのインターフェースを提供します。

Resource クラス

Resource クラスは、リポジトリ・オブジェクト `ResourceSpecifer.Resource` の単純化されたラッパーとして機能します。このクラスにより、オブジェクト固有の情報にアクセスすることができます。

このクラスには、リポジトリ・オブジェクトに関連する標準メタデータだけでなく、リポジトリ内のオブジェクト用に定義されたカスタム・メタデータの情報もすべて格納されます。61 ページの表 77 に、Resource クラスで使用可能なすべてのメソッドをリストします。

表 77. Resource クラスのメソッド	
メソッド名	説明
getAccessControllist	オブジェクトのセキュリティー権限の辞書を返します。この辞書には、キーとしてユーザー名が格納され、そのユーザーが持っている最上位の権限が格納されます。例: ユーザー Joe がリソース X に対する delete 権限を持っている場合、X を表すリソース・オブジェクトの getAccessControllist は { 'Joe': 'DELETE' } を返します。Web サービスの呼び出しで取得した 3 つの権限 (read、write、delete) をすべて返すわけではありません。
getOwner	オブジェクトの所有者の名前を文字列として返します。
getAuthor	オブジェクトの作成者の名前を文字列として返します。
getContentSize	オブジェクトのサイズを返します。
getCreatedBy	オブジェクトを作成したユーザーの名前を文字列として返します。
getCreationDate	オブジェクトの作成日を datetime オブジェクトとして返します。
getDescription	オブジェクトの説明をリストとして返します。
getDescriptionLanguage	オブジェクトの言語をリストとして返します。
getExpirationDate	オブジェクトの有効期限を datetime オブジェクトとして返します。
isExpired	指定したオブジェクトが有効期限切れかどうかを示します。
getMimeType	オブジェクトの MIME タイプを文字列として返します。
getModificationDate	オブジェクトの最終変更日を datetime オブジェクトとして返します。
getObjectCreationDate	オブジェクトの作成日を datetime オブジェクトとして返します。
getObjectLastModifiedBy	オブジェクトを最後に変更したユーザーを文字列として返します。
getObjectLastModifiedDate	オブジェクトの最終変更日を datetime オブジェクトとして返します。
getResourceID	オブジェクトのリソース ID を文字列として返します。
getResourcePath	指定したオブジェクトのパスを文字列として返します。
getTitle	オブジェクトのタイトルを文字列として返します。
getTopicList	オブジェクトのトピック・リストを返します。
getVersionMarker	オブジェクトのバージョンを文字列として返します。
getVersionLabel	オブジェクトのラベルを文字列として返します。
getCustomMetadata	オブジェクトに関連するすべてのカスタム・プロパティを辞書として返します。
getKeywordList	オブジェクトに関連付けられたキーワードのリストを返します。

IdentificationSpecifier クラス

このクラスは、リポジトリ・オブジェクト `IdentificationSpecifier` の単純化されたラッパーとして機能します。このクラスにより、オブジェクトの ID 固有のデータにアクセスすることができます。

62 ページの表 78 に、`IdentificationSpecifier` クラスで使用可能なすべてのメソッドをリストします。

表 78. <code>IdentificationSpecifier</code> クラスのメソッド	
メソッド名	説明
<code>getIdentifier</code>	オブジェクトの ID 値を文字列として返します
<code>getVersionMarker</code>	オブジェクトのバージョンを文字列として返します
<code>getVersionLabel</code>	オブジェクト・バージョンに適用されたラベルを文字列として返します

PageResult クラス

この `PageResult` クラスは、検索結果のコンテナとして機能します。検索結果の個々のヒットは、`PageResult` オブジェクトの各行に対応します。

例えば、検索結果として 4 つのリソースが返された場合は、`PageResult` オブジェクトに 4 つの行が格納されます。62 ページの表 79 に、`PageResult` クラスで使用可能なすべてのメソッドを示します。

表 79. <code>PageResult</code> クラスのメソッド	
メソッド名	説明
<code>getRows</code>	<code>SearchRow</code> オブジェクトのリストを返します。詳しくは、62 ページの『 SearchRow クラス 』のトピックを参照してください。

SearchRow クラス

`SearchRow` クラスは、個々の検索結果に関するオブジェクト・レベルの情報のコンテナとして機能します。このクラスのメソッドを使用して、オブジェクトに関するメタデータにアクセスすることができます。

62 ページの表 80 に、`SearchRow` クラスで使用可能なすべてのメソッドをリストします。

表 80. <code>SearchRow</code> クラスのメソッド	
メソッド名	説明
<code>getTitle</code>	ファイルまたはフォルダーの名前を返します。
<code>getAuthor</code>	ファイルまたはフォルダーの作成者を返します。
<code>getMimeType</code>	ファイルの MIME タイプを返します。
<code>getObjectLastModifiedBy</code>	ファイルまたはフォルダーを最後に変更したユーザーを返します。
<code>getModified</code>	ファイルまたはフォルダーが最後に変更された日時を返します。
<code>getFolderPath</code>	ファイルまたはフォルダーの場所を返します。
<code>getFolder</code>	ファイルまたはフォルダーの親フォルダーの名前を返します。
<code>getParentURI</code>	親のオブジェクト URI を返します。
<code>getTopic</code>	ファイルまたはフォルダーに関連付けられたトピックを返します。

表 80. SearchRow クラスのメソッド (続き)	
メソッド名	説明
getChildRow	SearchChildRow オブジェクトのリストを返します (詳しくは以下のセクションを参照してください)

オブジェクトのバージョン・レベルの情報にアクセスするには、getChildRow メソッドを使用してください。このメソッドは、オブジェクトのバージョンに対応する下位の行を返します。

SearchChildRow クラス

SearchChildRow クラスは、個々の検索結果に関するバージョン・レベルの情報のコンテナーとして機能します。このクラスのメソッドを使用して、オブジェクト・バージョンに関するメタデータにアクセスすることができます。

63 ページの表 81 に、SearchChildRow クラスで使用可能なすべてのメソッドをリストします。

表 81. SearchChildRow クラスのメソッド	
メソッド名	説明
getExpirationDate	ファイルまたはフォルダーの有効期限を返します。
getKeyword	ファイルまたはフォルダーのバージョンに関連付けられたキーワードを返します。
getVersionLabel	ファイルまたはフォルダーのバージョン・ラベルを返します。
getDescription	ファイルまたはフォルダーの説明を返します。
getLanguage	言語を返します。
getVersionCreationDate	ファイルまたはフォルダーが作成された日時を返します。
getVersionMarker	ファイルまたはフォルダーのバージョン・マーカを返します。
getUri	ファイルまたはフォルダーのオブジェクト URI を返します。

プロセス管理 API

プロセス管理スクリプトには、ジョブを処理する機能が組み込まれています。以下の機能を備えています。

- ジョブの実行
- ジョブ履歴の取得
- ジョブ詳細の取得

このセクションでは、リポジトリに格納されているジョブの処理で使用する PESImpl のメソッドについて説明します。すべてのメソッドについて、詳細な構文の情報、例、予測されるメッセージを記載しています。

メソッド

以下の各セクションでは、IBM スポンサー コラボレーションおよびデプロイメント・サービス でサポートされるすべてのプロセス管理スクリプトのメソッドを示します。

注: リポジトリ内のファイルやフォルダーのパスが必要なメソッドでは、パスとオブジェクト URI のいずれかを使用することができます。オブジェクト URI を調べるには、IBM SPSS デプロイメント・マネージャー でオブジェクトのプロパティを表示します。

cancelJob メソッド

実行中のジョブをキャンセルします。

```
cancelJob(executionId)
```

表 82. cancelJob の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
executionId	必須	ストリング	ジョブの実行 ID	0a58c33d002ce9080000010e0ccf7b01800e

表 83. cancelJob の戻り値

タイプ	説明
Boolean	ジョブがキャンセルされた場合に成功メッセージを返します。

例

以下の例では、*Reports* ジョブの実行を強制的に終了します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
execId = pesImpl.executeJob(source='/Demo/Jobs/Reports', notification = True, asynchronous=True)
print("Execution ID: ", execId)
status = pesImpl.cancelJob(execId)
print("Successful cancellation: ", status)

status = pesImpl.cancelJob(execId)
print "Successful cancellation: ", status
```

deleteJobExecutions メソッド

1 つ以上のジョブ実行を削除します。

```
deleteJobExecutions(executionId)
```

表 84. deleteJobExecutions の入力パラメーター

フィールド	使用	タイプ	説明	サンプルの値
executionId	必須	リスト	ジョブ実行 ID のリスト	[0a58c33d002ce9080000010e0ccf7b01800e, 0a59c33d002ce9080060010e0cdf7b01802r]

表 85. deleteJobExecutions の戻り値

タイプ	説明
Boolean	True または False。メソッドが正常に実行されたかどうかを示します。

例

以下の例では、*Reports* ジョブの実行を削除します。

```
from pes.util.PESExceptions import *
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
executions = pesImpl.getJobExecutionList(source="/Demo/Jobs/Reports")
execRows = executions.getRows()

# Get the execution ID from the execution history
deleteList = []
for exrow in execRows :
    uuid = exrow.getEventObjId()
    deleteList.append(uuid)
```

```
if len(deleteList) != 0:
    print print('Deleting ',len(deleteList) , ' histories')
    pesImpl.deleteJobExecutions(deleteList)
```

executeJob メソッド

渡されたパラメーターに基づいて、ジョブを同期的または非同期的に実行します。同期実行の場合は、ジョブが完了するまでメソッドが返されません。非同期実行の場合は、ジョブの開始後にメソッドが返されます。

```
executeJob(source,notification,asynchronous)
```

表 86. executeJob の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
source	必須	ストリング	アップロードするファイルの(ローカル・ファイル・システム上の)完全修飾パス。	"C:\Temp\Temp.txt"
notification	オプション	ブール値	ジョブを実行する際に通知するかどうかを指定します。デフォルトは False です。	True または False
asynchronous	オプション	ブール値	ジョブを非同期的に実行するかどうかを指定します。デフォルトは False です。	True または False

表 87. executeJob の戻り値

タイプ	説明
String	ジョブの実行を表す一意の識別子。この識別子を使用して特定のジョブ実行を参照します。

例

以下の例では、Reports ジョブの非同期的な実行を通知付きで開始します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[AES]KrY+KL10Yo406545tgGsYQ==", host = "localhost", port = "8080",
    appserver_type = "websphere")
execId = pesImpl.executeJob(source='/Demo/Jobs/Reports', notification = True, asynchronous=True)
print("Execution ID: ",execId)
```

getJobExecutionDetails メソッド

ジョブ・ステップや繰り返しなど、特定のジョブの実行の詳細をリストします。

```
getJobExecutionDetails(executionId,log,target)
```

表 88. getJobExecutionDetails の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
executionId	必須	ストリング	ジョブの実行 ID。	0a58c33d002ce9080000010e0ccf7b01800e
log	オプション	ブール値	ジョブ・ログをインラインで表示するかどうかを指定します。	True または False

表 88. `getJobExecutionDetails` の入力パラメーター (続き)

フィールド	用途	タイプ	説明	サンプルの値
ターゲット	オプション	ストリング	ログの保存先 (ローカル・ファイル・システム上)。 <code>--log</code> パラメーターとともに指定する必要があります。	"c:\logs"

表 89. `getJobExecutionDetails` の戻り値

タイプ	説明
<code>jobExecutionDetails</code>	ジョブ実行に関する詳細。 詳しくは、68 ページの『 jobExecutionDetails クラス 』のトピックを参照してください。

例

以下の例では、識別子が `execId` であるジョブ実行のジョブ・ステップ実行に関する情報を取得し、各ステップの結果をコンソールに送信します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
execDetails = pesImpl.getJobExecutionDetails(executionId=execId)
print("Job ID: ", execDetails.getUUID())
print("Event ID: ", execDetails.getEventUUID())
print("Started: ", execDetails.getStartDateTime())
print("Ended: ", execDetails.getEndDateTime())
for step in execDetails.getJobStepDetails():
    print("Step ID: ", step.getEventUUID())
    print("Step Name: ", step.getEventName())
    print("Started: ", step.getStartDateTime())
    print("Ended: ", step.getEndDateTime())
    print("Success: ", step.getExecutionSuccess())
```

getJobExecutionList メソッド

現在実行中のすべてのジョブと完了したジョブを含め、特定のジョブのすべてのバージョンについて、そのジョブの実行をリストします。

```
getJobExecutionList(source)
```

表 90. `getJobExecutionList` の入力パラメーター

フィールド	用途	タイプ	説明	サンプルの値
<code>source</code>	必須	ストリング	リポジトリ内のジョブの完全修飾パス。	"/testJob"

表 91. `getJobExecutionList` の戻り値

タイプ	説明
<code>PageResult</code>	ジョブの実行のリストを格納するコンテナ。 詳しくは、67 ページの『 PageResult クラス 』を参照してください。

例

以下の例では、`Reports` ジョブの実行を取得し、それぞれの実行に関する情報をコンソールに送信します。

```
from pes.api.PESImpl import PESImpl
pesImpl = PESImpl(user = "admin", password = "[{AES}KrY+KL10Yo406545tgGsYQ==]", host = "localhost", port = "8080",
appserver_type = "websphere")
```

```

executions = pesImpl.getJobExecutionList(source="/Demo/Jobs/Reports")
execRows = executions.getRows()
if execRows:
    for exrow in execRows:
        print("Job Path: ", exrow.getPath())
        print("Object ID: ", exrow.getObjId())
        print("Event ID: ", exrow.getEventObjId())
        print("Version ", exrow.getVersionMarker())
        print("Started: ", exrow.getEventStartDateTime())
        print("Ended: ", exrow.getEventEndDateTime())

```

ラッパー・クラス

PESImpl API には、プロセス管理のメソッドによって呼び出された Web サービスから返されるオブジェクトのラッパーとして機能するクラスが含まれています。これらのラッパー・クラスは、メソッドから返された情報を表示するためのインターフェースを提供します。

PageResult クラス

この PageResult クラスは、ジョブ実行結果のコンテナとして機能します。このクラスにより、ジョブ実行固有のデータを取得することができます。

個々のジョブ実行は、PageResult オブジェクトの各行に対応します。例えば、ジョブを 4 回実行すると、4 つの行を持つ PageResult オブジェクトが生成されます。67 ページの表 92 に、PageResult クラスで使用可能なすべてのメソッドをリストします。

表 92. PageResult クラスのメソッド	
メソッド名	説明
getRows	Row オブジェクトのリストを返します。各オブジェクトが、1 つのジョブの 1 つの実行を表します。詳しくは、67 ページの『Row クラス』のトピックを参照してください。

Row クラス

Row クラスは、ジョブ実行に関するジョブ・レベルの情報のコンテナとして機能します。このクラスのメソッドを使用して、ジョブ実行に関するメタデータにアクセスすることができます。

67 ページの表 93 に、Row クラスで使用可能なすべてのメソッドをリストします。

表 93. Row クラスのメソッド	
メソッド名	説明
getObjId	ジョブの実行 ID を返します。
getPath	ジョブのパスを返します。
getVersionMarker	実行されたジョブのバージョン・マーカを返します。
getVersionLabel	実行されたジョブのバージョン・ラベルを返します。
getEventObjId	実行されたジョブのイベント ID を返します。
getEventState	実行中のジョブの状態を返します。
getEventCompletionCode	ジョブの完了コードを返します。
getEventStartDateTime	ジョブの開始日時を返します。
getEventEndDateTime	ジョブの終了日時を返します。
getQueuedDateTime	ジョブがキューに格納された日時を返します。

jobExecutionDetails クラス

このクラスは `getJobExecutionDetails` メソッドから返されます。このクラスには、ジョブの実行の詳細が格納され、ジョブの各ステップに関する情報を提供する `jobStepExecution` オブジェクトのリストが格納されます。

68 ページの表 94 に、`jobExecutionDetails` クラスで使用可能なすべてのメソッドをリストします。

表 94. <code>jobExecutionDetails</code> クラスのメソッド	
メソッド名	説明
<code>getJobStepDetails</code>	<code>jobStepExecutionDetails</code> オブジェクトのリストを返します。詳しくは、68 ページの『 jobStepExecutionDetails クラス 』のトピックを参照してください。
<code>getArtifactLocation</code>	ジョブの成果物の場所のリストを返します。
<code>getCompletionCode</code>	ジョブ実行の完了コードを返します。
<code>getEndDateTime</code>	ジョブ実行の終了日時を返します。
<code>getEventName</code>	ジョブ実行のイベント名を返します。
<code>getEventUUID</code>	ジョブ実行のイベント ID を返します。
<code>getExecutionState</code>	ジョブ実行の実行状態を返します。
<code>getExecutionSuccess</code>	ジョブ実行の成功または失敗のステータスを返します。
<code>getExecutionWarning</code>	警告が発生したかどうかを示します。
<code>getLog</code>	生成されたログを (文字列として) 返します。
<code>getNotificationEnabled</code>	E メール通知が有効かどうかを示します。
<code>getQueuedDateTime</code>	ジョブ実行がキューに格納された日時を返します。
<code>getStartDateTime</code>	ジョブ実行の開始日時を返します。
<code>getUserName</code>	ジョブを実行したユーザーの名前を返します。
<code>getUUID</code>	ジョブの実行 ID を返します。

jobStepExecutionDetails クラス

このクラスには、ジョブ・ステップの実行の詳細と、`jobStepChildExecutionDetails` オブジェクトのリストが格納されます。このクラスには `ExecutionDetails` オブジェクトが含まれ、メソッドの呼び出しはすべてこのオブジェクトに委任されます。

68 ページの表 95 に、`jobStepExecutionDetails` クラスで使用可能なすべてのメソッドをリストします。

表 95. <code>jobStepExecutionDetails</code> クラスのメソッド	
メソッド名	説明
<code>getJobStepChldExecutionList</code>	<code>jobStepChildExecutionDetails</code> オブジェクトのリストを返します。詳しくは、69 ページの『 jobStepChildExecutionDetails クラス 』のトピックを参照してください。
<code>getArtifactLocation</code>	ジョブ・ステップ成果物の場所のリストを返します。
<code>getCompletionCode</code>	ジョブ・ステップの完了コードを返します。

表 95. *jobStepExecutionDetails* クラスのメソッド (続き)

メソッド名	説明
<code>getEndDateTime</code>	ジョブ・ステップの終了日時を返します。
<code>getEventName</code>	ジョブ・ステップのイベント名を返します。
<code>getEventUUID</code>	ジョブ・ステップのイベント ID を返します。
<code>getExecutionState</code>	ジョブ・ステップの実行状態を返します。
<code>getExecutionSuccess</code>	ジョブ・ステップの成功または失敗のステータスを返します。
<code>getExecutionWarning</code>	警告が発生したかどうかを示します。
<code>getLog</code>	生成されたログを (文字列として) 返します。
<code>getNotificationEnabled</code>	E メール通知が有効かどうかを示します。
<code>getQueuedDateTime</code>	ジョブ・ステップがキューに格納された日時を返します。
<code>getStartDateTime</code>	ジョブ・ステップの開始日時を返します。
<code>getUserName</code>	ジョブ・ステップを実行したユーザーの名前を返します。
<code>getUUID</code>	ジョブ・ステップの実行 ID を返します。

jobStepChildExecutionDetails クラス

`jobStepChildExecutionDetails` クラスは、個々のジョブ・ステップの子実行のコンテナとして機能します。例えば、繰り返しのレポート・ジョブ・ステップは、ステップの反復ごとに子実行を生成します。このクラスのメソッドを使用して、子実行に関するメタデータにアクセスすることができます。

69 ページの表 96 に、`jobStepChildExecutionDetails` クラスで使用可能なすべてのメソッドをリストします。

表 96. *jobStepChildExecutionDetails* クラスのメソッド

メソッド名	説明
<code>getArtifactLocation</code>	子実行の成果物の場所のリストを返します。
<code>getCompletionCode</code>	子実行の完了コードを返します。
<code>getEndDateTime</code>	子実行の終了日時を返します。
<code>getEventName</code>	子実行のイベント名を返します。
<code>getEventUUID</code>	子実行のイベント ID を返します。
<code>getExecutionState</code>	子実行の実行状態を返します。
<code>getExecutionSuccess</code>	子実行が成功したか失敗したかを返します。
<code>getExecutionWarning</code>	警告が発生したかどうかを示します。
<code>getLog</code>	生成されたログを (文字列として) 返します。
<code>getNotificationEnabled</code>	E メール通知が有効かどうかを示します。
<code>getQueuedDateTime</code>	子実行がキューに格納された日時を返します。
<code>getStartDateTime</code>	子実行の開始日時を返します。
<code>getUserName</code>	子実行を実行したユーザーの名前を返します。
<code>getUUID</code>	子実行の実行 ID を返します。

サンプル・スクリプト

PESImpl クラスの使用法を示すサンプル・スクリプトは、以下のディレクトリーにインストールされます。

<installation location>/samples

これらのスクリプトは、以下のようなさまざまなタスクを実行します。

- 有効期限が切れた項目を IBM スポス コラボレーションおよびデプロイメント・サービス・リポジトリーから削除する。
- 有効期限が切れた送信済みの成果物を削除する。
- ジョブ履歴を削除する。

IBM SPSS デプロイメント・マネージャー の一般ジョブ・ステップからスクリプトを起動して、リポジトリーの保守タスクを実行することができます。

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。この資料の他の言語版を IBM から入手できる場合があります。ただし、これを入手するには、本製品または当該言語版製品を所有している必要がある場合があります。

IBM は、このドキュメントで説明されている製品、サービス、または機能を、他の国では提供していない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒 103-8510

東京都中央区日本橋箱崎町 19 番 21 号

日本アイ・ビー・エム株式会社

法務・知的財産

US-->

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

<!--Intellectual Property Licensing

<!-- Legal and Intellectual Property Law -->

日本 アイ・ビー・エム株式会社

103-8510

Tokyo 103-8510, Japan-->

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

〒 103-8510

東京都中央区日本橋箱崎町 19 番 21 号

日本アイ・ビー・エム株式会社

法務・知的財産

US-->

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

記載されている性能データとお客様事例は、例として示す目的でのみ提供されています。実際の結果は特定の構成や稼働条件によって異なります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM はこれらの製品をテストしていないため、IBM 以外の製品に関連するパフォーマンス、互換性、またはその他のクレームの正確性を確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

IBM の将来の方向性および指針に関する記述は、予告なく変更または撤回される場合があります。これらは目標および目的を提示するものにすぎません。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名前はすべて架空のものであり、類似する個人や企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態提供されるものであり、いかなる保証も提供されません。IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品（「ソフトウェア・オファリング」）では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie などの各種テクノロジーの使用について詳しくは、『IBM オンラインでのプライバシー・ステートメントのハイライト』（<http://www.ibm.com/privacy/jp/ja/>）、『IBM オンラインでのプライバシー・ステートメント』（<http://www.ibm.com/privacy/details/jp/ja/>）の『クッキー、ウェブ・ビーコン、その他のテクノロジー』というタイトルのセクション、および『IBM Software Products and Software-as-a-Service Privacy Statement』（<http://www.ibm.com/software/info/product-privacy>）を参照してください。

商標

IBM、IBM ロゴ、および ibm.com は、世界の多くの国で登録された International Business Machines Corporation の商標です。世界中の多くの国で登録されています。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、<http://www.ibm.com/legal/copytrade.shtml> をご覧ください。

Adobe、Adobe ロゴ、PostScript、PostScript ロゴは、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Centrino、Intel Centrino ロゴ、Celeron、Xeon、Intel SpeedStep、Itanium、および Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。

