目次

改訂情報

変更年月日	変更内容
2013-07-01	初版
2014-01-01	第２版　下記を追加・変更しました ▪ 各項目に利用可能なバージョン情報について追記 ▪ 「*概要*」に「*インポート時のデータ検証について*」を追加 ▪ 「*オプション*」にデータ検証オプションの説明を追加 ▪ 「*インポート* 」に「*インポートの依存関係* 」を追加
2014-05-01	第３版　下記を追加・変更しました ▪ 「*オプション* 」の説明を変更 ▪ 「*インポート* 」に「*注意事項* 」を追加
2014-08-01	第４版　下記を追加・変更しました ▪ 「 *概要* 」に ロールの削除について追記

はじめに

本書の目的

本書ではロール情報のインポート・エクスポート機能の詳細について説明します。

説明範囲は以下のとおりです。

- ロール情報のインポート・エクスポートの概要
- ロール情報のインポート・エクスポートファイルのデータフォーマット
- ロール情報のインポート・エクスポートの実行方法
- ロール情報のインポート・エクスポートの実行オプション

対象読者

本書では次の利用者を対象としています。

- intra-mart Accel Platform のロールを管理する運用担当者
- ロール情報のインポート・エクスポート機能を利用したアプリケーションを開発する開発者

本書の構成

- *概要*

  ロール情報のインポート・エクスポートで扱う情報について説明します。

- *ファイルフォーマット*

  インポートファイルのデータフォーマットについて説明します。

- *インポート*

  インポートの処理について説明します。更新モードについても説明します。

- *エクスポート*

  エクスポートの処理について説明します。

- *オプション*

  インポート・エクスポート実行時に指定可能なオプションについて説明します。

- *実行方法*

  インポート・エクスポートの実行方法について説明します。

概要

ロールのインポート・エクスポート機能では、XML形式でロール情報のインポート・エクスポートを行います。
ロール情報とは、ロールに関連する以下の情報です。

- *ロール情報*
- *ロール表示名情報*
- *ロール親子関係*

ロールの削除を行う機能は提供されていません。
ただし、既存のロールを置き換えることは可能です。
詳しくは、「 *インポート* 」-「 *更新モード* 」-「 *replace* 」を参照してください。

インポート・エクスポートで扱う情報

扱う情報の制限については「*インポート時のデータ検証について*」を参照してください。

ロール情報

ロールに対しての設定値を保持します。

ロール情報を構成する項目は以下の通りです。

項目	対応するテーブル名	対応するカラム名	説明
ロールID	b_m_role_b	role_id	システム内部で使用されるロールの識別子です。
ロール名	b_m_role_b	role_name	ロールの名称です。
カテゴリ	b_m_role_b	category	ロールのカテゴライズを行うための値です。
説明	b_m_role_b	notes	ロールに設定されている説明です。

ロール表示名情報

ロールに設定された表示名です。
ロケールごとの表示名を保持します。

ロール表示名情報を構成する項目は以下の通りです。

項目	対応するテーブル名	対応するカラム名	説明
表示名	b_m_role_i	display_name	ロールに設定されている言語別の表示名です。 ロケールごとに設定可能です。

ロール親子関係

ロールに設定されたロールの親子情報です。

ロールの親子関係を構成する項目は以下の通りです。

項目	対応するテーブル名	対応するカラム名	説明
親ロールID	b_m_role_inclusion_b	parent_role_id	ロールに設定されている親ロールIDです。
子ロールID	b_m_role_inclusion_b	child_role_id	ロールに設定されている子ロールIDです。

コラム
ロール親子関係 を登録した場合ロールサマリ(b_m_role_summary)にもレコードが作成されます。

インポート時のデータ検証について

インポート実行時には登録・更新処理を実行する前にデータの検証を行います。データ検証にて条件に一致しない場合はインポートに失敗します。
データ検証の実行の有無は「*オプション*」にて変更可能です。
データ検証内容については以下の通りです。

ロール情報の検証

- ロールID

  ロールIDは 1 文字以上の文字列を必ず指定する必要があります。
  ロールIDは半角英字、数字、アンダースコア、ハイフン、アットマーク、ドット、プラス、または、エクスクラメーションに限定されます。
  ロールIDは20文字以内である必要があります。

- ロール名

  ロール名は 1 文字以上の文字列を必ず指定する必要があります。
  ロール名は半角英字、数字、アンダースコア、ハイフン、アットマーク、ドット、プラス、または、エクスクラメーションに限定されます。
  ロール名は50文字以内である必要があります。

- カテゴリ

  カテゴリは半角英字、数字、アンダースコア、ハイフン、アットマーク、ドット、プラス、または、エクスクラメーションに限定されます。
  カテゴリは255文字以内である必要があります。

- 説明

  説明は63文字以内である必要があります。

ロール表示名情報の検証

- ロケールID

  ロケールIDは20文字以内である必要があります。

- 表示名

  表示名は63文字以内である必要があります。
  テナントロケールに対しての表示名が必ず含まれている必要があります。

ロール親子関係の検証

- 親ロールID

ロール情報として登録されているロールのロールIDを指定する必要があります。
同一インポートファイルに存在するロールも指定できます。

- 子ロールID

  ロール情報として登録されているロールのロールIDを指定する必要があります。
  同一インポートファイルに存在するロールも指定できます。

ファイルフォーマット

項目

この章では、インポート・エクスポート機能で利用するファイルのフォーマットについて説明します。

XMLファイルフォーマット

<role-data> タグ内に 1 つのロールに関連する情報をすべてを記述します。

ロール情報 を <role-data> タグに記述します。
id 属性にロールIDを、name 属性にロール名を記述します。
タグ内には、カテゴリ、備考を記述します。

ロール表示名情報 を <display-names> タグに記述します。
設定数の数だけ <display-name> タグを記述します。
locale 属性にロケールIDを、タグ内には、ロケールIDに応じたロールの表示名を記述します。

ロール親子関係 の親ロール情報を <parent-roles> タグに記述します。
設定数の数だけ <parent-role> タグを記述します。
id 属性に親ロールIDを記述します。

ロール親子関係 の子ロール情報を <sub-roles> タグに記述します。
設定数の数だけ <sub-role> タグを記述します。
id 属性に子ロールIDを記述します。

ファイルフォーマットの詳細については「*ロールインポートデータ定義書*」を参照してください。

以下はXMLファイルの例です。

```
<root xmlns="http://intra-mart.co.jp/system/admin/role/role-data">
  <role-data name="role-1" id="role-1">
    <description>Top role.</description>
    <display-names>
      <display-name locale="ja">ロール 1 </display-name>
      <display-name locale="en">role 1</display-name>
    </display-names>
    <parent-roles />
    <sub-roles>
      <sub-role id="role-2" />
    </sub-roles>
  </role-data>
  <role-data name="role-2" id="role-2">
    <description>Sub role.</description>
    <display-names>
      <display-name locale="ja">ロール 2 </display-name>
      <display-name locale="en">role 2</display-name>
    </display-names>
    <parent-roles>
      <parent-role id="role-1" />
    </parent-roles>
    <sub-roles>
      <sub-role id="role-3" />
    </sub-roles>
  </role-data>
  <role-data name="role-3" id="role-3">
    <description>Sub role.</description>
    <display-names>
      <display-name locale="ja">ロール 3 </display-name>
      <display-name locale="en">role 3</display-name>
    </display-names>
    <parent-roles>
      <parent-role id="role-2" />
    </parent-roles>
    <sub-roles />
  </role-data>
</root>
```

コラム

parent-roles タグと sub-roles タグの記述について

どちらか一方のみ指定することで親子関係が構築されます。
上記の例ではロールID role-2 の sub-roles タグにロールID role-3 を指定し、
ロールID role-3 の parent-roles タグにロールID role-2 を指定していますが、
親となるロールと子となるロールの両方に親子関係を指定する必要はありません。

互換性

intra-mart Accel Platform で利用するインポートファイルは、 intra-mart WebPlatform/AppFramework で利用するインポートファイルとは互換性はありません。
intra-mart WebPlatform/AppFramework で利用していたインポートファイルをそのまま利用したい場合、互換機能を利用してください。
互換機能を利用するためには、「互換ガイド」を参照してください。

インポート

項目

ロールのインポートはXML形式で行うことが可能です。
マスタ情報の新規登録、更新を行うことができます。

この章では、インポートがどのように行われるかを説明します。
また *更新モード* による更新方法の違いについて説明します。

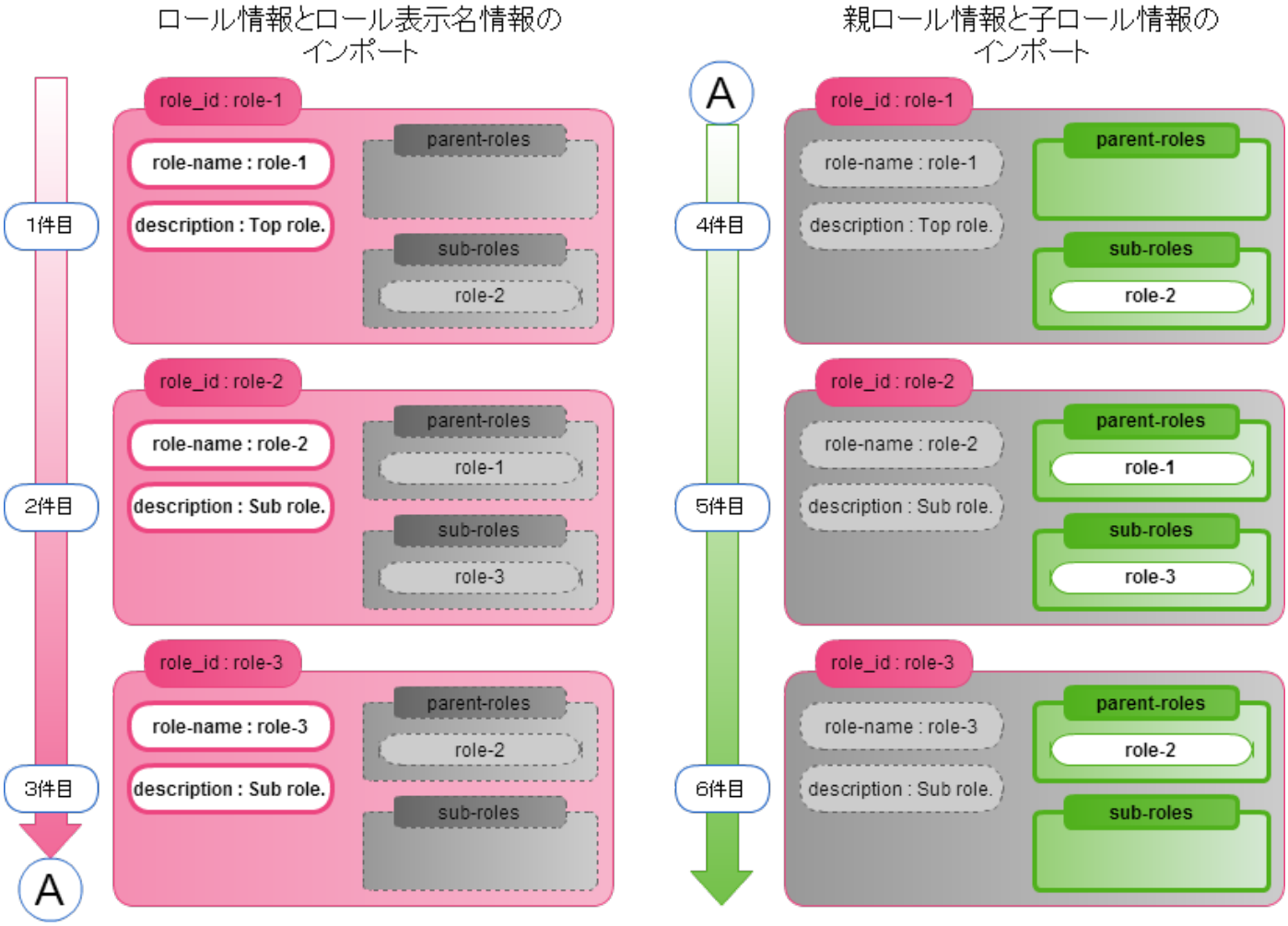
XML

ロールのインポートは `<role-data>` タグの内容を以下の２つに分割してインポートします。

- ロールの情報（*ロール情報* と *ロール表示名情報*）
- *ロール親子関係*

これはロールの親子関係を更新する際に、親となるロールまたは子となるロールがインポートファイルの後方に記述されている場合に、存在しない（インポートが完了していない）ロールを親子関係に追加するとエラーとなる現象を回避するためです。
ファイル内すべての *ロール情報* と *ロール表示名情報* をインポート後に *ロール親子関係* のインポートを行います。
このため、インポート結果の件数はインポートファイルに記述されているロール数の２倍です。

以下は３つのロールをインポートする場合の処理フローです。



ファイルのフォーマットについては「*XMLファイルフォーマット*」を参照してください。

また、インポート時の動作をオプションとして指定が可能です。
インポート時に利用可能なオプションについては、「*インポートオプション*」を参照してください。

> **注意**
>
> *ロール親子関係* のインポートでは親ロール情報と子ロール情報の両方が記述されている場合や
> 親ロール情報と子ロール情報の両方が記述されていない場合も、インポート結果の件数の内の 1 件として扱います。

更新モード

更新モードを利用することで、インポートファイルのデータがデータベース上に存在する場合（更新を行う場合）の
データの更新方法を変更できます。
更新モードには *merge* と *replace* が提供されています。
`<role-data>` タグに `update-mode` 属性を指定することでモードを設定します。
特に指定していない場合は、*merge* モードで動作します。

merge

インポートファイルのデータとデータベース上のデータをマージして更新します。
インポートファイルに存在しない項目は既存のデータをそのまま設定されます。

replace

インポートファイルのデータに存在しない情報は未設定の値（デフォルト値）で更新します。
インポートファイルに存在しない項目は未設定です。

インポートの依存関係

ロールをインポートする際にあらかじめ登録が必要なデータは、以下の通りです。

- サブロールに対する親となるロール

> **コラム**
>
> 親ロールが他のインポートファイルに記述されている場合は、親ロールを先にインポートしてください。

注意事項

親子関係の整合性

親ロール情報と子ロール情報が循環するようなデータのインポートを行うとエラーが発生します。
親子関係の整合性がとれたデータをインポートしてください。

ロール名の重複

ロール名はロール毎にユニークな値となる必要があります。
インポートを行うロールに指定されているロール名が既に存在する場合、エラーが発生します。
（ただし、更新を行おうとしているロールのロール名を以前と同じ値で更新する場合は、この限りではありません。）

エラーが発生する例

	ロールID	ロール名
登録済みのロール	sample_role_id_1	sample_role_name
インポートするロール	sample_role_id_2	sample_role_name

エクスポート

項目
- XML

ロールのエクスポートはXML形式で行うことが可能です。
すべてのマスタ情報を出力できます。

この章では、エクスポートがどのように行われるかを説明します。

XML

すべてのロールとそのロールに関連する情報をXML形式でファイルに出力します。
子ロール情報は出力されません。*ロール親子関係* は「親ロール情報」のみ出力されます。
ファイルのフォーマットについては「*XMLファイルフォーマット*」を参照してください。

また、エクスポート時の動作をオプションとして指定が可能です。
エクスポート時に利用可能なオプションについては、「*インポートオプション*」を参照してください。

オプション

項目
- インポートオプション
- エクスポートオプション

インポート・エクスポートで扱うファイルなどの設定や、インポート・エクスポート処理の動作を変更するためのオプションが用意されています。

この章では、インポート・エクスポート時に使用できるオプションについて説明します。

インポートオプション

この項では、インポートを行う際に使用可能なオプションについて説明します。

設定可能項目一覧

名前	キー名	型	デフォルト値	導入バージョン
エンコーディング	encoding	文字列	UTF-8	2012 Autumn
ファイルパス	file	文字列	（なし）	2012 Autumn
コミット件数	commit-count	数値	0	2012 Autumn
XML検証フラグ	validate-xml	真偽値	true	2012 Autumn
データ検証フラグ	validate-data	真偽値	true	2013 Winter

エンコーディング

キー名 encoding

インポートするXMLファイルの文字エンコーディングを指定します。

ファイルパス

キー名 file

インポートするXMLファイルのパス（パブリックストレージのルートからの相対パス）を指定します。

コミット件数

キー名 commit-count

インポート処理で、コミットを行うまでのデータ件数を指定します。
コミット件数に「0」（デフォルト値）が指定された場合は、インポート処理が完了するまでコミットが行われません。

注意

commit-count を指定した場合、インポート実行元で管理しているトランザクションがコミットされる可能性があります。

XML検証フラグ

キー名 validate-xml

インポートするXMLファイルの構文を検証するかどうかを指定します。

指定する値	説明
true	XML構文の検証を行います。（デフォルト値）
false	XML構文の検証を行いません。

データ検証フラグ

キー名 validate-data

インポートするデータの検証を行うかどうかを指定します。

指定する値	説明
true	データの検証を行います。（デフォルト値）
false	データの検証を行いません。

エクスポートオプション

この項では、エクスポートを行う際に使用可能なオプションについて説明します。

設定可能項目一覧

名前	キー名	型	デフォルト値	導入バージョン
エンコーディング	encoding	文字列	UTF-8	2012 Autumn
ファイルパス	file	文字列	（なし）	2012 Autumn
XML整形フラグ	format-xml	真偽値	false	2012 Autumn
ルートタグ名	root-tag-name	文字列	root	2012 Autumn
書き込み件数	flush-count	数値	5000	2012 Autumn

エンコーディング

キー名 encoding

エクスポートするXMLファイルの文字エンコーディングを指定します。

ファイルパス

キー名 file

エクスポートするXMLファイルのパス（パブリックストレージのルートからの相対パス）を指定します。

XML整形フラグ

キー名 format-xml

エクスポートするXMLファイルを整形するかどうかを指定します。

指定する値	説明
true	XMLの整形を行います。
false	XMLの整形を行いません。（デフォルト値）

ルートタグ名

キー名 root-tag-name

エクスポートするXMLファイルのルートタグ名を指定します。

書き込み件数

キー名 flush-count

エクスポートするXMLファイルに一度に書き込むデータ件数を指定します。

実行方法

項目
- ジョブスケジューラを利用する
- Javaから実行する
- スクリプト開発モデルプログラムから実行する

この項では、インポート・エクスポートを実行する手段を紹介します。

> **注意**
> インポートしたデータをエクスポートする場合、またはエクスポートしたデータをインポートする場合は、「エンコーディング」などの対応するオプションは同じ値を指定する必要があります。

ジョブスケジューラを利用する

ジョブスケジューラの機能を利用してインポート・エクスポートを実行する方法を紹介します。
ジョブスケジューラの詳細については「ジョブスケジューラ仕様書」を参照してください。
intra-mart Accel Platform ではロールのインポート・エクスポートを行うためのジョブ・ジョブネットを提供しています。
この項では、intra-mart Accel Platform が標準で提供しているロールのインポート・エクスポートを行うジョブ・ジョブネットの情報を紹介します。

ジョブ

- ロールインポート

ジョブカテゴリ	テナントマスタ > インポート
ジョブID	role-import
ジョブ名	ロールインポート

- ロールエクスポート

ジョブカテゴリ	テナントマスタ > エクスポート
ジョブID	role-export
ジョブ名	ロールエクスポート

ジョブネット

- ロールインポート

ジョブネットカテゴリ	テナントマスタ > インポート
ジョブネットID	role-import-jobnet
ジョブネット名	ロールインポート

- ロールエクスポート

ジョブネットカテゴリ	テナントマスタ > エクスポート
ジョブネットID	role-export-jobnet
ジョブネット名	ロールエクスポート

コラム

ジョブスケジューラ利用時のオプションについて

ジョブスケジューラを利用してインポート・エクスポートを実行する場合は、
ジョブ・ジョブネットのパラメータに「*オプション*」を指定します。

コラム

ジョブスケジューラ利用時のトランザクション管理について

ジョブスケジューラを利用したインポートでは、オプション commit-count を指定しない場合、インポート処理が完了後に一括してコミットを行います。
必要に応じてオプション commit-count の値を変更してご利用ください。

Javaから実行する

JavaのAPIを利用してインポート・エクスポートを実行する方法を紹介します。

インポート

DataImportExecutor#importData(String, InputStream, Map) を利用してインポートを行います。

- 完全修飾クラス名

  jp.co.intra_mart.foundation.data.importer.DataImportExecutor

第１引数にはインポータIDを指定します。インポータIDは以下を利用します。

jp.co.intra_mart.import.StandardRoleXmlImporter

第２引数にはインポート元を InputStream で指定します。

第３引数にはインポートオプションを Map<String, Object> で指定します。
詳細は、「*インポートオプション*」を参照してください。

DataImportExecutor の詳細については「DataImportExecutorクラスのAPIリスト」を参照してください。

以下はインポートを行うサンプルプログラムです。

```
package sample;

import java.io.IOException;
import java.io.InputStream;
import java.util.HashMap;
import java.util.Map;

import jp.co.intra_mart.foundation.data.OptionKeyName;
import jp.co.intra_mart.foundation.data.exception.DataImporterException;
import jp.co.intra_mart.foundation.data.importer.DataImportExecutor;
import jp.co.intra_mart.foundation.service.client.file.PublicStorage;

/**
 * ロールのインポートを行うクラスです。
 */
public class RoleImporter {

    private static final String IMPORTER_ID = "jp.co.intra_mart.import.StandardRoleXmlImporter";

    /**
     * ロールのインポートを行います。
     * @throws DataImporterException インポートで何らかの例外が発生した場合。
     */
    public void doImport() throws DataImporterException {
        final DataImportExecutor executor = new DataImportExecutor();
        final Map<String, Object> options = new HashMap<String, Object>();
        options.put(OptionKeyName.ENCODING.value(), "UTF-8");
        options.put(OptionKeyName.VALIDATE_XML.value(), true);
        options.put(OptionKeyName.COMMIT_COUNT.value(), 100);
        // パブリックストレージ直下のrole.xmlを選択
        final PublicStorage storage = new PublicStorage("role.xml");
        try {
            // PublicStorageからInputStreamを取得
            final InputStream stream = storage.open();
            try {
                executor.importData(IMPORTER_ID, stream, options);
            } finally {
                stream.close();
            }
        } catch (final IOException e) {
            throw new DataImporterException(e);
        }
    }

}
```

> **注意**
>
> 第２引数の InputStream を指定した場合は、オプション file は利用できません。
> オプション file を利用したい場合は、第２引数に null を指定してください。

エクスポート

DataExportExecutor#exportData(String, OutputStream, Map) を利用してエクスポートを行います。

- 完全修飾クラス名

  jp.co.intra_mart.foundation.data.exporter.DataExportExecutor

第１引数にはエクスポータIDを指定します。エクスポータIDは以下を利用します。

jp.co.intra_mart.export.StandardRoleXmlExporter

第２引数にはエクスポート先を OutputStream で指定します。

第３引数にはエクスポートオプションを Map<String, Object> で指定します。
詳細は、「*エクスポートオプション*」を参照してください。

DataExportExecutor の詳細については「DataExportExecutorクラスのAPIリスト」を参照してください。

以下はエクスポートを行うサンプルプログラムです。

```
package sample;

import java.io.IOException;
import java.io.OutputStream;
import java.util.HashMap;
import java.util.Map;

import jp.co.intra_mart.foundation.data.OptionKeyName;
import jp.co.intra_mart.foundation.data.exception.DataExporterException;
import jp.co.intra_mart.foundation.data.exporter.DataExportExecutor;
import jp.co.intra_mart.foundation.service.client.file.PublicStorage;

/**
 * ロールのエクスポートを行うクラスです。
 */
public class RoleExporter {

   private static final String EXPORTER_ID = "jp.co.intra_mart.export.StandardRoleXmlExporter";

   /**
    * ロールのエクスポートを行います。
    * @throws DataExporterException エクスポートで何らかの例外が発生した場合。
    */
   public void doExport() throws DataExporterException {
      final DataExportExecutor executor = new DataExportExecutor();
      final Map<String, Object> options = new HashMap<String, Object>();
      options.put(OptionKeyName.ENCODING.value(), "UTF-8");
      options.put(OptionKeyName.FORMAT_XML.value(), false);
      options.put(OptionKeyName.FETCH_COUNT.value(), 10);
      // パブリックストレージ直下のrole.xmlを選択
      final PublicStorage storage = new PublicStorage("role.xml");
      try {
         // PublicStorageからOutputStreamを取得
         final OutputStream stream = storage.create();
         try {
            executor.exportData(EXPORTER_ID, stream, options);
         } finally {
            stream.close();
         }
      } catch (final IOException e) {
         throw new DataExporterException(e);
      }
   }

}
```

> **注意**
>
> 第２引数の OutputStream を指定した場合は、オプション file は利用できません。
> オプション file を利用したい場合は、第２引数に null を指定してください。

スクリプト開発モデルプログラムから実行する

スクリプト開発モデルのAPIを利用してインポート・エクスポートを実行する方法を紹介します。

インポート

DataImportExecutor#importData(String, ByteReader, Object) を利用してインポートを行います。

第１引数にはインポータIDを指定します。インポータIDは以下を利用します。

jp.co.intra_mart.import.StandardRoleXmlImporter

第２引数にはインポート元を ByteReader で指定します。

第３引数にはインポートオプションをObject形式で指定します。
詳細は、「*インポートオプション*」を参照してください。

DataImportExecutor の詳細については「DataImportExecutorオブジェクトのAPIリスト」を参照してください。

以下はインポートを行うサンプルプログラムです。

```
var IMPORTER_ID = 'jp.co.intra_mart.import.StandardRoleXmlImporter';

/**
 * ロールのインポートを行います。
 */
function doImport() {
    var executor = new DataImportExecutor();
    var options = {
        'encoding': 'UTF-8',
        'validate-xml': true,
        'commit-count': 100
    };
    // パブリックストレージ直下のrole.xmlを選択
    var storage = new PublicStorage('role.xml');
    // ファイルからデータをインポート
    storage.openAsBinary(function(reader, error) {
        if (error) {
            // ファイルの読み込みに失敗 -> 例外処理
            Logger.getLogger().error(error.message);
            return;
        }

        var result = executor.importData(IMPORTER_ID, reader, options);
        if (result.error) {
            // インポート失敗 -> 例外処理
            Logger.getLogger().error(result.errorMessage);
        }
    });
}
```

> **注意**
>
> 第２引数の ByteReader を指定した場合は、オプション file は利用できません。
> オプション file を利用したい場合は、第２引数に null を指定してください。

エクスポート

DataExportExecutor#importData(String, ByteWriter, Object) を利用してエクスポートを行います。

第１引数にはエクスポータIDを指定します。エクスポータIDは以下を利用します。

`jp.co.intra_mart.export.StandardAccountXmlExporter`

第２引数にはエクスポート元を `ByteWriter` で指定します。

第３引数にはエクスポートオプションをObject形式で指定します。
詳細は、「*エクスポートオプション*」を参照してください。

`DataExportExecutor` の詳細については「DataExportExecutorオブジェクトのAPIリスト」を参照してください。

以下はインポートを行うサンプルプログラムです。

```
var EXPORTER_ID = 'jp.co.intra_mart.export.StandardRoleXmlExporter';

/**
 * ロールのエクスポートを行います。
 */
function doExport() {
   var executor = new DataExportExecutor();
   var options = {
      'encoding': 'UTF-8',
      'format-xml': false,
      'fetch-count': 10
   };
   // パブリックストレージ直下のrole.xmlを選択
   var storage = new PublicStorage('role.xml');
   // ファイルにデータをエクスポート
   storage.createAsBinary(function(writer, error) {
      if (error) {
         // ファイルの作成に失敗 -> 例外処理
         Logger.getLogger().error(error.message);
         return;
      }

      var result = executor.exportData(EXPORTER_ID, writer, options);
      if (result.error) {
         // エクスポートに失敗 -> 例外処理
         Logger.getLogger().error(result.errorMessage);
      }
   });
}
```

注意

第２引数の `ByteWriter` を指定した場合は、オプション `file` は利用できません。
オプション `file` を利用したい場合は、第２引数に `null` を指定してください。

項目

ロールインポートデータ定義書

ロールインポート・エクスポートで利用するデータの詳細はロールインポートデータ定義書として提供します。
ロールインポートデータ定義書は以下からダウンロードできます。

「**im_admin_role_import_export_definition.xls**」