オブジェクトストレージ SDK　Java　Version 1.0

Version	更新日	内容
1.0	2014 年 4 月 18 日	新規作成
1.1	2014 年 5 月 23 日	RiakCS のバージョンを 1.4.5 に修正

目次

1. 概要

本書は、IDC フロンティアの提供するオブジェクトストレージサービスの API を、Java から利用するオブジェクトストレージ SDK のスタートガイドです。

SDK を利用することで、オブジェクトストレージにバケットを作成、オブジェクトのアップロードなどの API 操作を Java からご利用いただけます。
このドキュメントでは API を呼び出すサンプルコードの掲載と、サンプルプログラムの実行方法の紹介を行っています。
オブジェクトストレージサービスは、IDC フロンティアセルフクラウドのアカウントをお持ちの方がご利用いただけます。

1.1. AWS SDK を継承

本書で紹介するオブジェクトストレージ SDK は、AWS が提供する S3 の SDK を継承して作成されております。
http://aws.amazon.com/jp/sdkforjava/
RiakCS にアクセスする RiakCSClient クラスは AmazonS3Client クラスを継承しています。

1.2. SDK の詳細な API

SDK 同梱の Javadoc をご覧ください。

SDK	SDK 内の Javadoc
オブジェクトストレージ SDK の Javadoc	idcf-cloudstorage-java-sdk-X.X.X/doc
S3 の Javadoc	idcf-cloudstorage-java-sdk-X.X.X/third-party/aws-java-sdk-1.6.1/documentation/javadoc

1.3. SDK での ACL の設定について

現在のオブジェクトストレージ SDK を使った ACL の設定では、バケット、オブジェクトに対して、特定のユーザーのみに限定しない ACL の設定ができます。(PublicRead、AuthenticatedRead など)

2. SDK から利用できる API

オブジェクトストレージサービスは RiakCS で構成されています。RiakCS の API のうち、バケット操作、オブジェクト操作の API をご利用いただけます。
RiakCS バージョン 1.4.5(2014 年 5 月 23 日現在)
http://docs.basho.com/riakcs/latest/references/apis/storage/s3/

オブジェクトストレージ SDK は、S3 の SDK 機能を利用して、API を呼び出します。現在動作を確認している API は下記の〇がついているものです。

#	レベル	API	SDK 対応	備考
1.	サービスレベル	バケットの一覧表示	〇	
2.	バケットレベル	バケット内のオブジェクトの一覧表示	〇	
3.		バケットの ACL を取得	〇	
4.		バケットのポリシーを取得	〇	
5.		バケットの作成	〇	
6.		バケットの ACL を設定	△	ユーザーを限定しない権限の設定が可能。ユーザー単位の権限設定に SDK が未対応。
7.		バケットのポリシーを設定	〇	
8.		バケットの削除	〇	
9.		バケットのポリシーを削除		

#	レベル	API	SDK 対応	備考
10.	オブジェクトレベル	オブジェクトのダウンロード	○	
11.		オブジェクトの ACL を取得	○	
12.		オブジェクトのコピー	×	
13.		オブジェクトの ACL を設定	△	ユーザーを限定しない権限の設定が可能。ユーザー単位の権限設定に SDK が未対応。
14.		オブジェクトのメタデータを取得	○	
15.		オブジェクトの削除	○	
16.		マルチパートアップロード	×	

3. SDK の構成

ディレクトリ・ファイル	内容
classes	コンパイルされたクラスを格納
doc	Javadoc を格納
src	ソースファイルを格納
target	src 内容ソースファイルをコンパイルし、jar ファイルにしたものを格納
third-party	SDK が使用するサードパーティー製ライブラリーを格納
.classpath	eclipse でのクラスパス設定（隠しファイル）
.gitignore	git 管理対象外ファイル（隠しファイル）
.project	eclipse 用プロジェクトファイル（隠しファイル）

4. SDK を利用するための準備

4.1. 必要な環境

Java 1.7. 45

Ant 1.8.4

Java のインストールされた実行環境をご準備ください。また Ant の実行環境があれば、Ant からサンプルプログラムの実行や、SDK を jar ファイル形式で出力、javadoc の出力ができます。

4.2. アクセスキー、シークレットキーの発行

オブジェクトストレージにアクセスする際に、アクセスキー、シークレットキーを利用します。

SDK ご利用の前に、オブジェクトストレージコントロールパネルで、アクセスキー、シークレットキーを発行してください。

5. サンプルプログラムの実行

※事前にアクセスキー、シークレットキーをご準備ください。(4.2 アクセスキー、シークレットキーの発行)
SDK を使ったバケット操作(作成、削除、バケット一覧の表示、ACL の設定)、オブジェクト操作(オブジェクトのアップロード、ダウンロード、削除、バケット内のオブジェクト一覧の表示、ACL の設定)のサンプルプログラムを SDK に同梱しています。

- サンプルプログラム ： src/sample/RiakCSSample.java

サンプルプログラム実行前に、事前に発行したアクセスキー、シークレットキーを src/RiakCSCredentials.properties に設定してください。

5.1. 実行方法

Ant で実行する場合、コマンドラインで SDK の src ディレクトリに移動し、ant コマンドを実行します。

```
$ ant
```

または、eclipse などの ide から sample.RiakCSSample.java の main メソッドを実行してください。third-party ディレクトリ内のライブラリを実行時のクラスパスに追加されている状態で実行してください。

6. SDK での API 操作

ここでは、RiakCSClient を使った API についてバケット操作、オブジェクト操作のサンプルコードを紹介します。 API の詳しい内容については Javadoc をご覧ください。
RiakCSClient は AWS SDK の提供する AmazonS3Client を継承しています。
RiakCSClient のオーバーライド、独自実装の機能は「doc/index.html」をご覧ください。

6.1. クライアント

API は RiakCSClient クラスのインスタンスで操作します。 RiakCSClient は API 操作のためのアクセスキー、シークレットキーを内部で参照します。 参照の方法はインスタンス化の種類により複数存在します。 ここではクラスパス上の RiakCSCredentials.properties からアクセスキー、シークレットキーを読み込みむ方法を紹介します。

```
RiakCSClient client = new RiakCSClient(new RiakCSClasspathCredentialsProvider());
```

6.2. バケット操作

6.2.1. バケットの作成

```
String bucketName = "sample";
client.createBucket(bucketName);
```

- 引数のバケット名は、すべてのバケットでユニークなものを指定する必要があります。
- バケット作成時の ACL は Private です。(下記「バケットの ACL を設定」参照)

6.2.2. バケットの ACL を設定

バケットに ACL を設定することでバケットを利用するユーザーを制限することができます。

認証なしでアクセス可能にする権限を設定する例

```
client.setBucketAcl(bucketName, CannedAccessControlList.PublicRead);
```

- 第 2 引数に指定できる ACL の定数の種類は、「6.4 バケット、オブジェクトの ACL」参照

6.2.3. バケットの ACL を取得

```
for (Grant g : client.getBucketAcl(bucketName).getGrants()) {
    System.out.println(g.getGrantee().getIdentifier());//権限の対象者
    System.out.println("    " + g.getPermission());//権限
    System.out.println("");
}
```

➢ 実行結果

```
id:http://acs.amazonaws.com/groups/global/AllUsers
  WRITE

id:http://acs.amazonaws.com/groups/global/AllUsers
  READ

id:fc58ae16c9d560af9209d717826266a83a87e82eb567868520fe171b54d59a33
  FULL_CONTROL
```

6.2.4. バケットの削除

```
client.deleteBucket(bucketName);
```

6.3. オブジェクト操作

6.3.1. オブジェクトのアップロード

```
File file = new File("/path/to/filename");
String key = "filename";
client.putObject(bucketName, key, file);
```

6.3.2. オブジェクトの取得

```
String key = "filename";
S3Object object = client.getObject(bucketName, key);
InputStream is = object.getObjectContent();
```

6.3.3. オブジェクトの ACL の設定

➢ アップロード時に設定

```
String key = "filename";
PutObjectRequest objectRequest = new PutObjectRequest(bucketName, key, new File("/path/to/filename"));
objectRequest.setCannedAcl(CannedAccessControlList.PublicRead);
client.putObject(objectRequest);
```

➢ すでにアップロードされているオブジェクトに設定

```
client.setObjectAcl(bucketName, key, CannedAccessControlList.PublicRead);
```

6.3.4. オブジェクトの ACL の取得

```
System.out.println("owner:" + acl.getOwner().getDisplayName());
for(Grant g : acl.getGrants()){
   System.out.println(g.getGrantee().getIdentifier());
   System.out.println("   " +  g.getPermission());
}
```

➢ 実行結果

```
owner:user1
fc58ae16c9d560af9209d717826266a83a87e82eb567868520fe171b54d59a33
  FULL_CONTROL
```

6.3.5. オブジェクトのメタデータの取得

```
S3Object object = client.getObject(new GetObjectRequest(bucketName, key));
System.out.println("Content-Type: "  + object.getObjectMetadata().getContentType());
```

➢ 実行結果

```
Content-Type: text/plain
```

6.3.6. オブジェクトの削除

```
client.deleteObject(bucketName, key);
```

6.4. バケット、オブジェクトの ACL 用定数

➢ com.amazonaws.services.s3.model.CannedAccessControlList の定数

定数	効果
Private	バケット、オブジェクト作成時の初期値。所有者以外がアクセスできない。
PublicRead	バケット、オブジェクト所有者はフルコントロールのアクセスができる。 全てのユーザーが認証なしで読み取り専用アクセスができる。
PublicReadWrite	全てのユーザーが認証なしで読み取り、書き込みのアクセスができる。
AuthenticatedRead	認証済みのユーザーが読み取専用アクセスができる。
BucketOwnerRead	バケット所有者が読み取専用のアクセスができる。
BucketOwnerFullControl	バケット所有者がフルコントロールのアクセスができる。

※フルコントロール：作成、削除、読み取、書き込み、ACL の設定など