# BL-08 ファイル操作機能

## ■ 概要

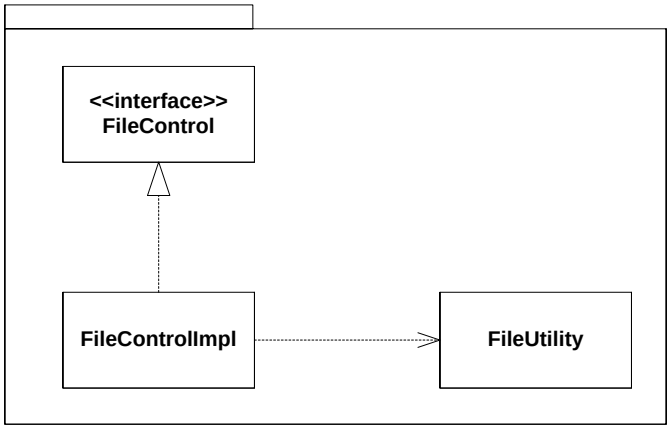
### ◆ 機能概要

- ファイル操作機能は以下の機能を提供する。
  - ファイル名変更
    変更元ファイルと変更先のファイルを指定して、ファイル名の変更を行う。
  - ファイル移動
    ファイルの移動は「ファイル名変更」を利用し、パスを変更することにより実現する。
  - ファイルコピー
    コピー元ファイルとコピー先ファイルを指定して、ファイルのコピーを行う。
  - ファイル削除
    削除するファイルを指定して、ファイルの削除を行う。
  - ファイル結合
    指定されたファイル名のリストにあるファイルを結合する。

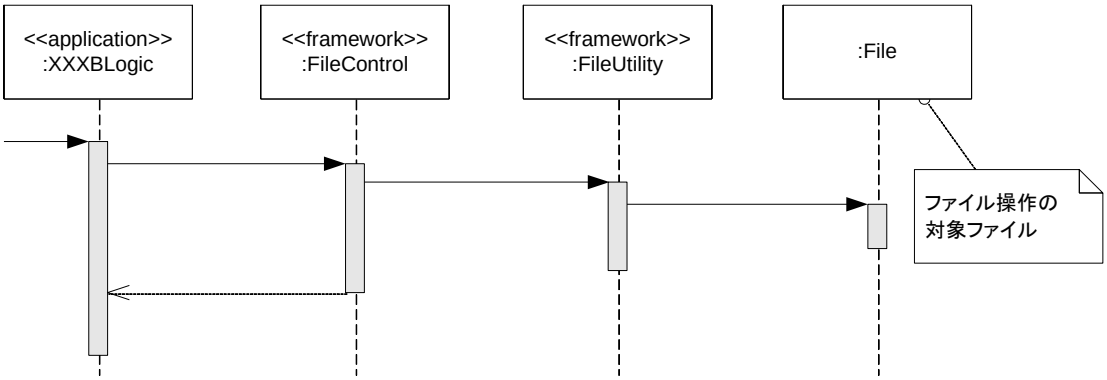
- 本機能は、TERASOLUNA Batch Framework for Java ver 2.x の『BC-02 ファイル操作機能』と同等である。

## ◆ 概念図

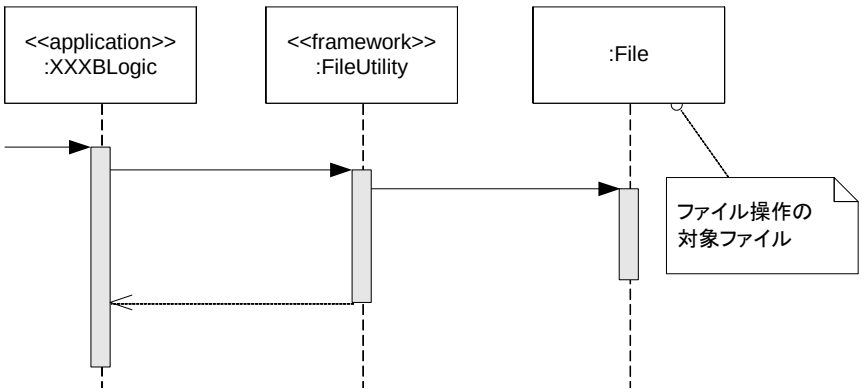
- ファイル操作機能のクラス図
  FileControl、FileControlImpl、FileUtility クラスの関連は下記のクラス図のとおりである。




- ファイル操作インタフェースを使用する場合




- ファイル操作ユーティリティクラスを直接使用する場合



## ◆ 解説

● ファイル操作インタフェース

ファイル操作の処理を提供するインタフェースとして、フレームワークでは FileControl インタフェースを提供する。また、デフォルト実装としてファイル操作ユーティリティクラスを使用する FileControlImpl クラスを提供する。

FileControlImpl クラスは、操作対象のファイルを絶対パスで指定するほか、属性にファイル操作を行う際に基準パス(※1)を持たせることで、相対パスで指定することもできる。相対パスで措定することにより、ファイルアクセス時に発生するファイルパスの環境依存の問題を回避することができる。ファイル操作を行うメソッドの引数には相対パス、または、絶対パスを設定する。ファイル操作時にエラーが発生した場合、フレームワークは非検査例外 FileException をスローする。

(※1)ファイル操作機能を使う際の基準となる位置を指す。基準パスを「/si1/」、相対パスを「chohyo/test.txt」とした場合、ファイルの絶対パスは「/si1/chohyo/test.txt」となる。

✧ ファイル操作インタフェース

| 項番 | インタフェース名 | 概要 |
|---|---|---|
| 1 | jp.terasoluna.fw.file.util.FileControl | ファイル操作用のインタフェース |

✧ ファイル操作インタフェースで提供する代表的なメソッド

| 項番 | メソッド | 概要 |
|---|---|---|
| 1 | renameFile(String srcFile, String newFile) | ファイル名の変更･ファイルの移動 |
| 2 | copyFile(String srcFile, String newFile) | ファイルのコピー |
| 3 | deleteFile(String srcFile) | ファイルの削除 |
| 4 | mergeFile(List<String> fileList, String newFile) | fileList にあるファイルの結合 |

✧ ファイル操作クラス

| 項番 | インタフェース名 | 概要 |
|---|---|---|
| 1 | jp.terasoluna.fw.file.util.FileControlImpl | ファイル操作用のインタフェースのデフォルト実装クラス |

- ファイル操作ユーティリティクラス

ファイル操作機能を実装するクラス。FileUtility のメソッドをビジネスロジックから直接利用することも可能である。FileUtility クラスで提供するメソッドは FileControl インタフェースで提供するものと同じになる。FileUtility クラスではファイル操作を行うメソッドの引数は絶対パスのみ設定可能である。ファイル操作時にエラーが発生した場合、フレームワークは非検査例外 FileException をスローする。

✧ ファイル操作ユーティリティクラス

| 項番 | メソッド | 概要 |
|---|---|---|
| 1 | jp.terasoluna.fw.file.util.FileUtility | ファイル操作ユーティリティクラス |

# ■ 使用方法

## ◆ コーディングポイント

- ファイル操作インタフェースを使用する場合
  - ファイル操作インタフェース実装クラスの Bean 定義
  - ビジネスロジックの実装

- ファイル操作ユーティリティクラスを直接使用する場合
  - ビジネスロジックの実装

- ファイル操作インタフェースを使用する場合
- ファイル操作インタフェース実装クラスの Bean 定義
  FileControlImpl の Bean を定義する。basePath 属性、checkFileExist 属性の設定ができる。

✧ Bean 定義ファイル定義例(commonContext.xml)

```
……
<bean id="fileControl"
      class="jp.terasoluna.fw.file.util.FileControlImpl">
      <property name="basePath" value="${basepath}" />
      <property name="checkFileExist" value="false" />
</bean>
……
```

FileControl インタフェースを実装するクラスをフレームワーク Bean 定義ファイルに定義する。プロパティに基準パスを設定すること。

操作後にできるファイルパスにファイルが存在する場合、処理を継続する(上書きする:true)か、例外を投げて停止する(false)かを決めるフラグ。

- ビジネスロジックの実装
  ビジネスロジックで FileControl インタフェース実装クラスを DI し使用する。

✧ ビジネスロジックの実装例(ファイルのコピー、移動、削除)

```
@Component
public class Sample001BLogic implements BLogic {
      @Inject
      FileControl fileControl = null;
      (…中略…)
      public int execute(BLogicParam arg0) {
            // ファイルのコピー(相対パスを設定する例)
            //   /si1/chohyo/testtxt を/si1/chohyo/testFile.txt にコピー。
            // 基準パスは「/si1/」
            fileControl.copyFile("chohyo/test.txt", "chohyo/testFile.txt");
            ……
            // ファイルの移動（相対パスを設定する例）
            //   /si1/chohyo/testFile.txt を/si1/output/testFile.txt に移動。
            // 基準パスは「/si1/」
            fileControl.renameFile("chohyo/testFile.txt", "output/testFile.txt");
            ……
            //ファイルの削除（相対パスを設定する例）
            //   /si1/chohyo/testFile.txt を削除。
            //基準パスは「/si1/」
            fileControl.deleteFile("chohyo/testFile.txt");
            ……
            // ファイルのコピー(絶対パスを設定する例)
            //   /si1/chohyo/test.txt を/si1/chohyo/testFile.txt にコピー。
            fileControl.copyFile("/si1/chohyo/test.txt", "/si1/chohyo/testFile.txt");
(…以下略…)
```

ファイル操作機能を利用するビジネスロジックは、@Inject を使用して FileControl インタフェース実装クラスをビジネスロジックに DI する。

各メソッドの引数はファイルの相対パス、もしくは絶対パスを記述する

✧ ビジネスロジックの実装例(ファイルの結合)

```
……
// ファイルの結合。
// 以下に挙げるファイルをリストに格納し、ファイルを
/si1/output/mergeFile.csv に統合。
//   /si1/chohyo/output001.csv
//   /si1/chohyo/output002.csv
//   /si1/chohyo/output003.csv
// 基準パスは「/si1/」
fileList.add("chohyo/output001.csv");
fileList.add("chohyo/output002.csv");
fileList.add("chohyo/output003.csv");
……
fileControl.mergeFile(fileList, "output/mergeFile.csv");
……
```

メソッドの 2 番目の引数はファイルの相対パス、もしくは絶対パスを記述する

- ファイル操作ユーティリティクラスを直接使用する場合
  ビジネスロジックで FileUtility の static メソッドを呼びだし、使用する。

➢ ビジネスロジックの実装

```
……
// ファイルのコピー。
//   /si1/chohyo/test.txt を/si1/chohyo/testFile.txt にコピー。
FileUtility.copyFile("/si1/chohyo/test.txt", "/si1/chohyo/testFile.txt");
……
// ファイルの移動。
//   /si1/chohyo/testFile.txt を/si1/output/testFile.txt に移動。
FileUtility.renameFile("/si1/chohyo/testFile.txt",
      "/si1/output/testFile.txt");
……
//ファイルの削除。    /si1/chohyo/testFile.txt を削除。
FileUtility.deleteFile("/si1/chohyo/testFile.txt");
……
```

各メソッドの引数はファイルの絶対パスを記述する

## ◆ 拡張ポイント

- なし。

## ■ 関連機能

- なし。

## ■ 使用例

- 機能網羅サンプル(terasoluna-batch-functionsample)
- チュートリアル(terasoluna-batch-tutorial)

## ■ 備考

- なし。