

TextPorter V5 Java インターフェース利用ガイド

初版 2010/03/05

1.1 版 2013/02/08

-d64 オプション、-Xss オプションの記述を追加。

1.2 版 2013/07/04

2013 年 2 月、Java 6 の公式サポートが終了したことに伴い、Java 6 の記述を Java 7 に変更。
Java 6 での動作保証はなくなり、動作保証は、Java 7 以上になりました。

1.3 版 2016/04/20

2015 年 4 月、Java 7 の公式サポートが終了したことに伴い、Java 7 の記述を Java 8 に変更。
Java 7 での動作保証はなくなり、動作保証は、Java 8 以上になりました。

1.4 版 2020/08/03

Java 11 以前のバージョンは、公式サポートが終了したことに伴い、記述を Java 11 に変更。
動作保証は、Java 11 になりました。次期 LTS(長期保証)の Java 17 が出荷されるまで、Java 11 が動作保証対象になります。LTS ではない Java 12, Java 13, Java 14, Java 15, Java 16 は動作保証ではありません。

1.5 版 2025/04/25

Java 18 からデフォルトの文字エンコーディングが、OS 依存のエンコーディングから、UTF-8 に変更になりました。Java 18 以降の Java、LTS の Java 21 や今後の LTS 使う場合、Windows では、API 名の末尾に W が付く、ワイド文字の API を使うようにしてください。
これに合わせて、サンプルの Test.java も変更しています。参考にしてください。

ディレクトリ/ファイル構成

- Window 系プラットフォーム

Lib¥

dmcjava.dll

dmc_java¥

apidoc.zip	Java インターフェース API 仕様のアーカイブ
apptest¥	サンプルソースコード
apptest.bat	サンプル実行用バッチファイル
dmcjava.jar	Java インターフェース jar
*.class	サンプルソースから作られたクラス

- Unix 系プラットフォーム

Lib/

libdmcjava.so.5.x	JNI(Java Native Interface)のファイル。 x は、TextPorter のマイナーバージョン番号。
libdmcjava.so.5	libdmcjava.so.5.x へのシンボリックリンク
libdmcjava.so	libdmcjava.so.5 へのシンボリックリンク

dmc_java/

apidoc.zip	Java インターフェース API 仕様のアーカイブ
apptest/	サンプルソースコード

appcmd.sh	サンプル実行用シェルスクリプトファイル
dmcjava.jar	Java インターフェース jar
*.class	サンプルソースから作られたクラス

動作環境の設定

TextPorter の環境設定

TextPorter の導入ガイドを参考に TextPorter をインストールしてください。

注：

すでに正常に TextPorter が使用できる状態であれば、不要です。

Java の環境設定

Java をインストールしてください。

注：

すでに正常に Java が使用できる状態であれば、不要です。

- 1) 各プラットフォーム用 JDK のインストール

各プラットフォーム用 JDK をインストールしてください。Java 11 の JDK を推奨。

- 2) 環境変数の設定

JDK をインストール後、JAVA_HOME を設定してください。

例

Unix で /usr/java/jdk11 にインストールした場合(bash の例)

```
export JAVA_HOME = /usr/java/jdk11
```

Windows で c:\jdk11 にインストールした場合

```
set JAVA_HOME=c:\jdk11
```

PATH に JAVA_HOME ディレクトリの bin(例 /usr/java/jdk11/bin や c:\jdk11\bin)を追加してください。

テスト実行手順

TextPorter と Java の環境設定を行った上で、dmc_java ディレクトリで、次のコマンドライン

Windows

```
java -cp dmcjava.jar;. Test 抽出元ファイル、出力先、オプション
```

Unix

```
java -cp dmcjava.jar;. Test 抽出元ファイル、出力先、オプション
```

などを入力して、テキストが抽出できれば、テスト成功です。

(app_*の使用方法とほぼ同じです。TextPorter のマニュアル「利用ガイド」をご参照ください)

例 Windows

```
java -cp dmcjava.jar;. Test d:\testdata\%* -t d:\outttx -p DMC_GETTEXT_OPT_OLE
```

例 Unix

```
java -cp dmcjava.jar;. Test ~/testdata/* -t ~/outttx -p DMC_GETTEXT_OPT_OLE
```

64bit の TextPorter を使う場合は、後述の「Java コマンドのオプション」を参照してください。

dmcjava.jar の扱い

dmcjava.dll や libdmcjava*.so.*については、TextPorter のインストールで、他の TextPorter の dll や so と同じところにあり、それらが正しくロードされ、実行できるように環境変数なども設定されているはずですから、省略し、dmcjava.jar の扱いについてののみ、述べます。

dmcjava.jar は、Java 仮想マシン(JVM)が jar をロードできるディレクトリであれば、どこにコピーしてもかまいません。ただし、そのディレクトリを JAR_DIR とすると、JAR_DIR¥dmcjava.jar や JAR_DIR/dmcjava.jar を、Java のやり方に従って、環境変数 CLASSPATH や Java の起動オプションで指定してください。

例

Java の起動オプションで指定する例。

JAR_DIR は、お客様の環境に合わせて変更してください。

Windows の例

```
java -cp JAR_DIR¥dmcjava.jar;その他の jar Java のプログラム
```

Unix の例

```
java -cp JAR_DIR/dmcjava.jar:その他の jar Java のプログラム
```

Java コマンドのオプション

■ 64bit の指定

通常、64bit プラットフォーム用の Java コマンドは、-d64 オプションなしで 64bit 動作します。

しかし、プラットフォームにより、64bit の TextPorter を使う場合に、java コマンドに -d64 オプションが必要な場合があります。

プラットフォームによっては、64bit の Java がない場合や 32bit プラットフォームの Java コマンドに、-d64 オプションがない場合があります。その場合は、64bit の動作はできません。

■ スタック領域の指定

複数のスレッドを使ったり、アーカイブファイル、OLE に埋め込んだファイル、PDF やメールに添付したファイルの抽出時には、スタック領域が足りなくなり、クラッシュすることがあります。

java コマンドの -Xss オプションで十分なスタックサイズを指定してください。

詳しい java コマンドのオプションの解説は、

<https://docs.oracle.com/en/java/javase/11/tools/java.html#GUID-3B1CE181-CD30-4178-9602-230B800D4FAE>

を参照してください。

インターフェース仕様

dmc_java¥apidoc.zip を展開してご覧下さい。

