

# 新規コンポーネント作成手順

= 音を鳴らすコンポーネントを例にして =

平成 19 年 10 月 12 日: MZ Platform.2.0



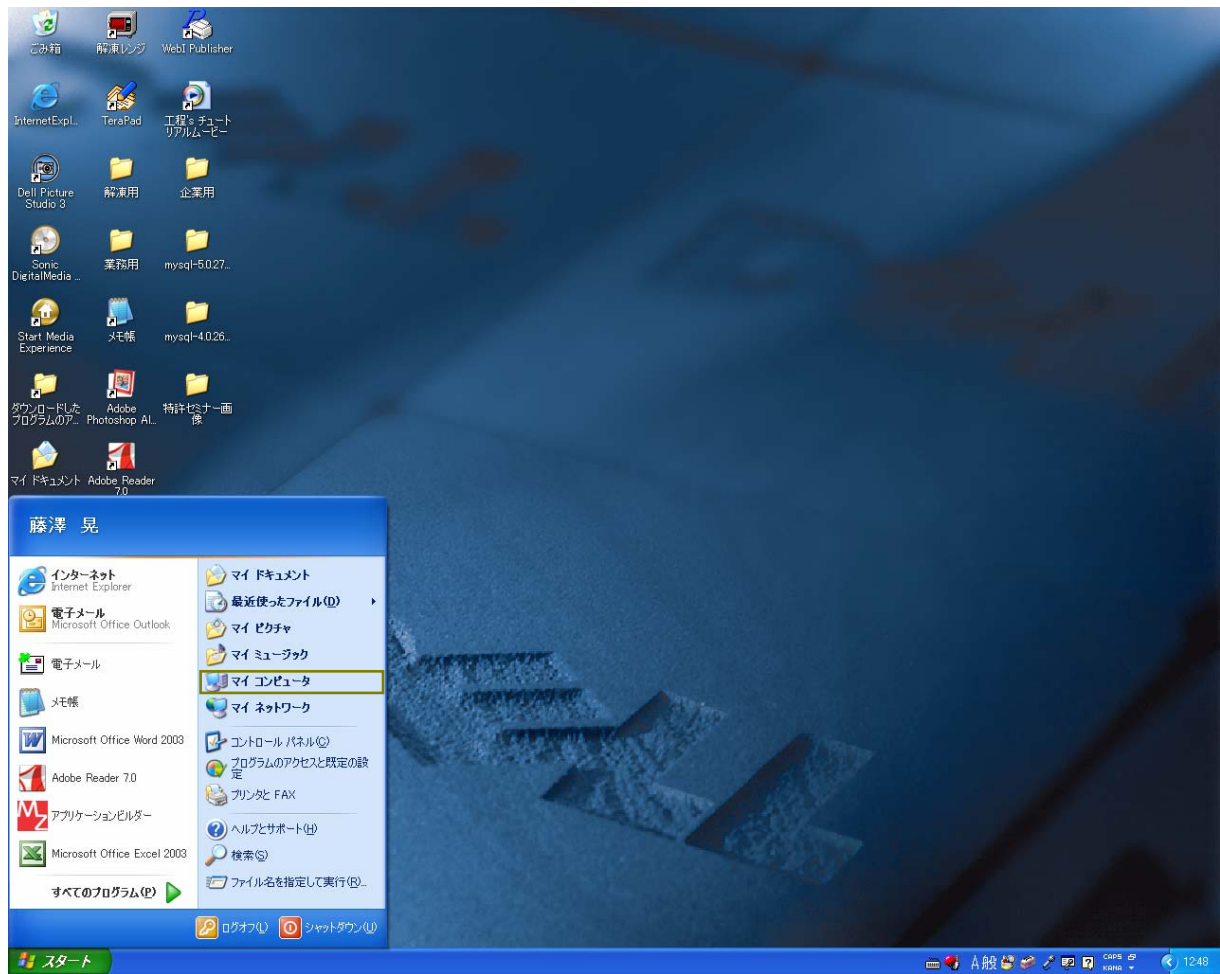
独立行政法人  
産業技術総合研究所

＝目次＝

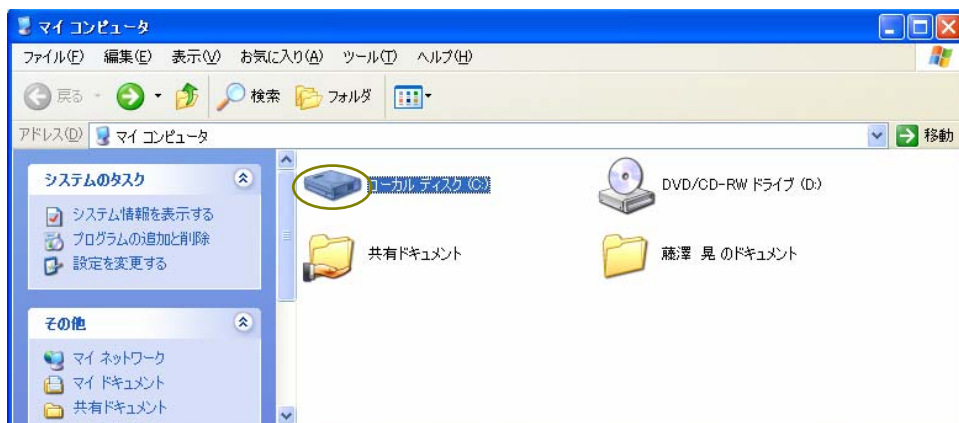
手順 1	サンプルプログラム格納場所への移動.....	3
手順 2	サンプルプログラム名の変更.....	7
2-1	ファイル名の変更.....	7
2-2	プログラムコードの編集.....	11
2-3	「SAMPLE_EN.PROPERTIES」ファイルの修正.....	17
2-4	「SAMPLES_JA.PROPERTIES」ファイルの修正.....	20
手順 3	「BUILD.BAT」ファイルの編集.....	23
3-1	ファイルのコピー.....	23
3-2	バッチファイルのプログラム編集.....	27
手順 4	バッチファイルの実行.....	34
手順 5	「SAMPLE.JAR」ファイルの確認.....	37
手順 6	コンポーネント登録手続き（その 1）.....	40
手順 7	コンポーネント登録手続き（その 2）.....	42
手順 8	コンポーネント一覧に登録されているかの確認.....	44

## 手順1 サンプルプログラム格納場所への移動

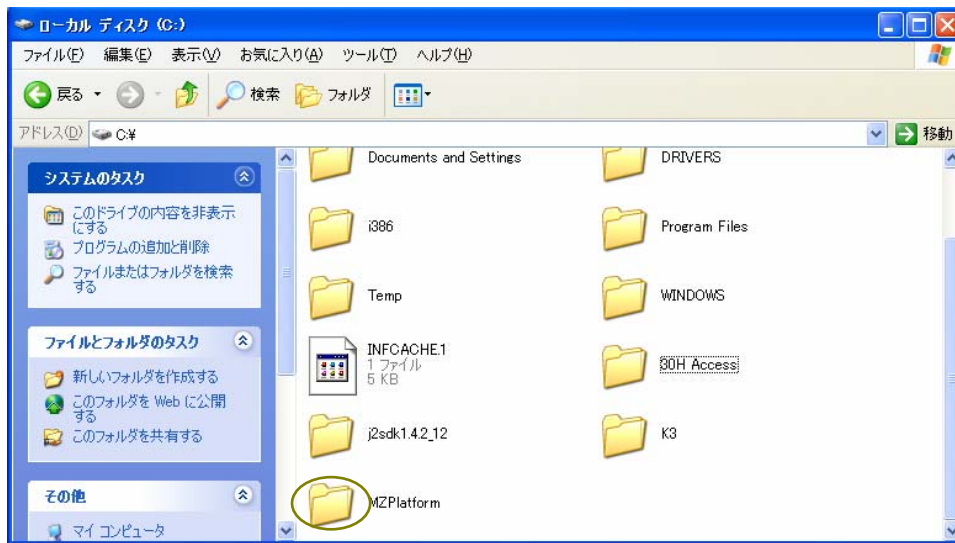
「スタート」 → 「マイコンピュータ」とたどり、左クリックします。



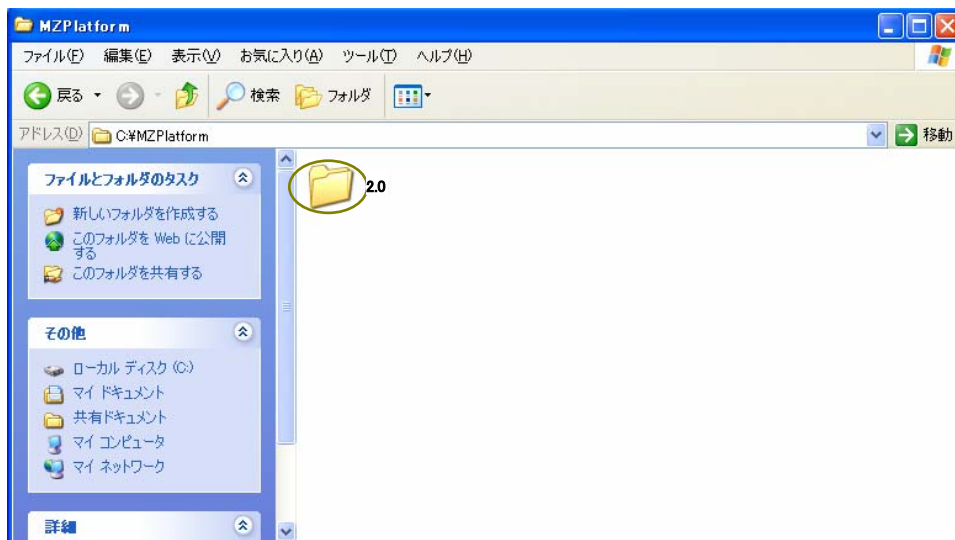
「マイコンピュータ」画面が表示されます。「ローカルディスク(C:)」アイコンをダブルクリックします。



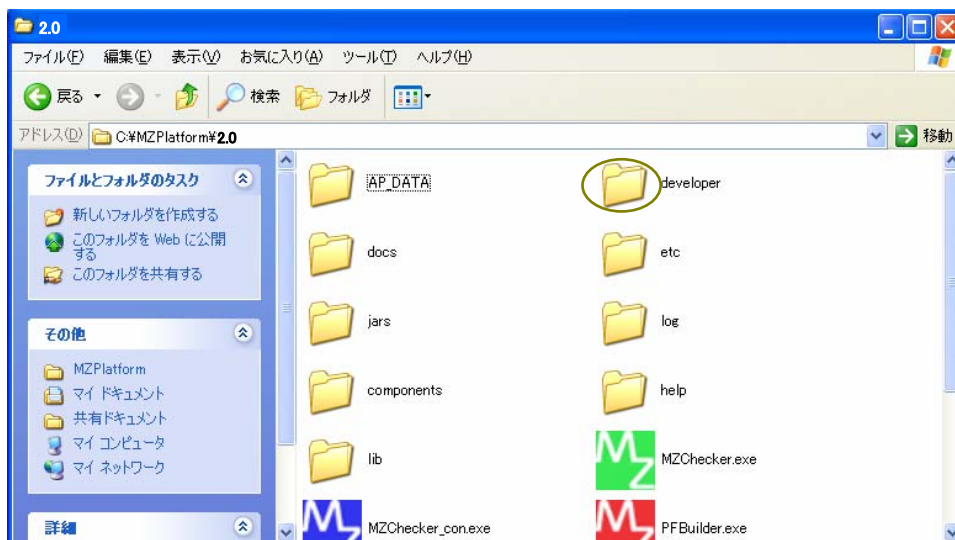
「ローカルディスク(C:)」画面が表示されます。「MZPlatform」フォルダのアイコンをダブルクリックします。



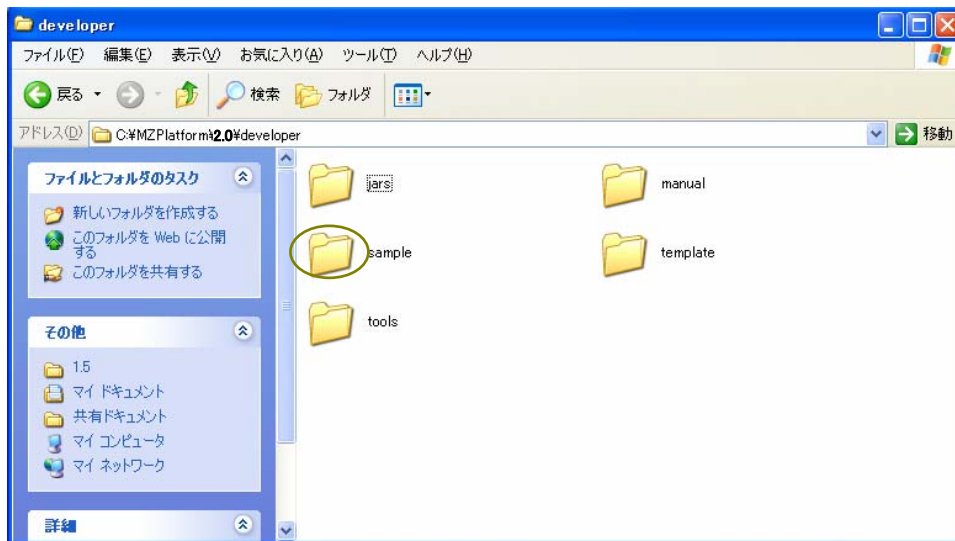
「MZPlatform」画面が表示されます。「2.0」フォルダのアイコンをダブルクリックします。



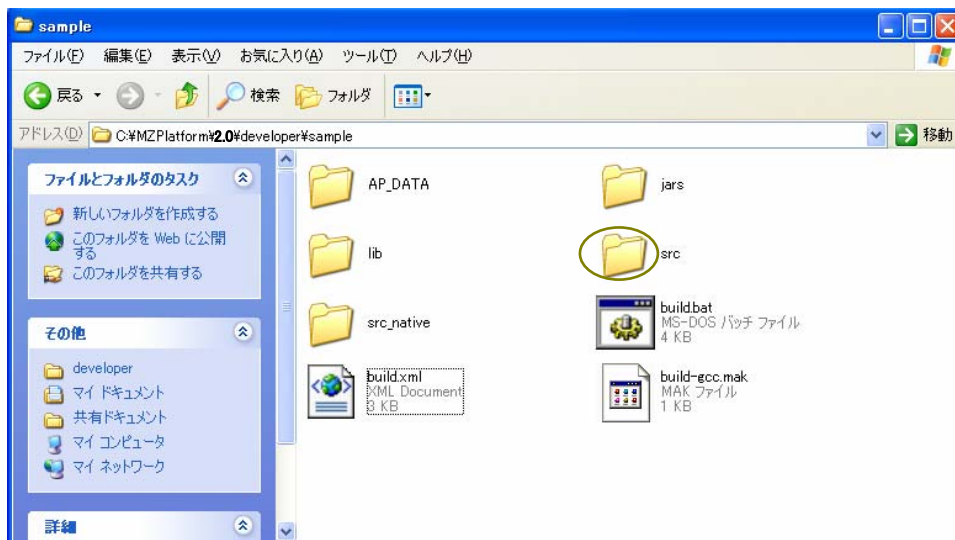
「2.0」画面が表示されます。「developer」フォルダのアイコンをダブルクリックします。



「developer」画面が表示されます。「sample」フォルダのアイコンをダブルクリックします。



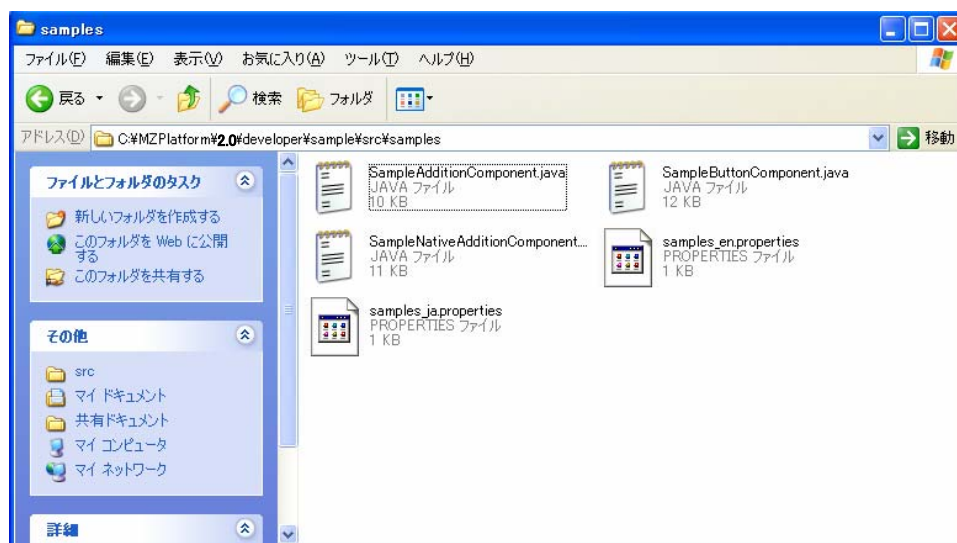
「sample」画面が表示されます。「src」フォルダのアイコンをダブルクリックします。



「src」画面が表示されます。「samples」フォルダのアイコンをダブルクリックします。



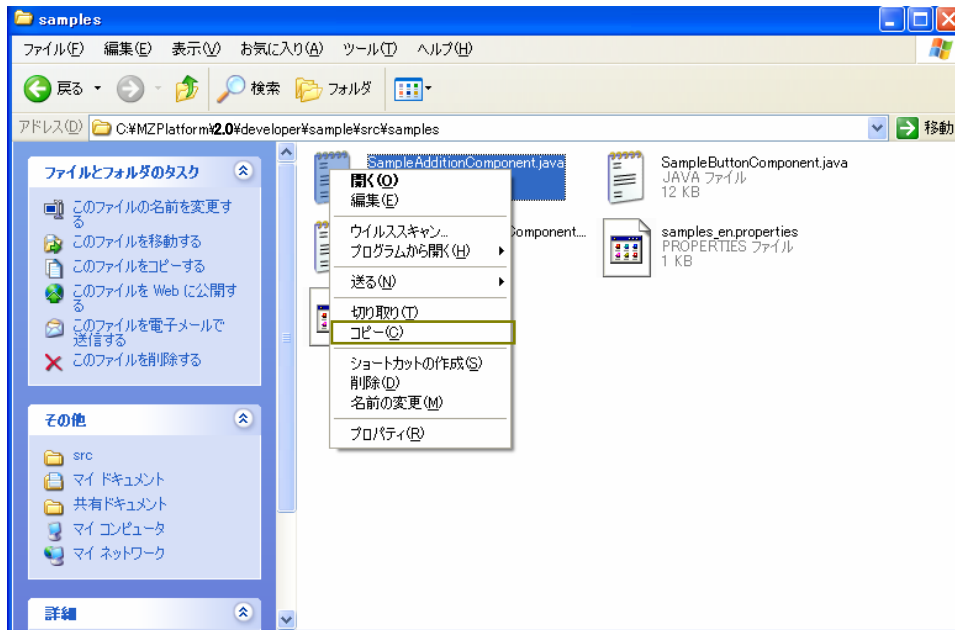
「samples」画面が表示されます。



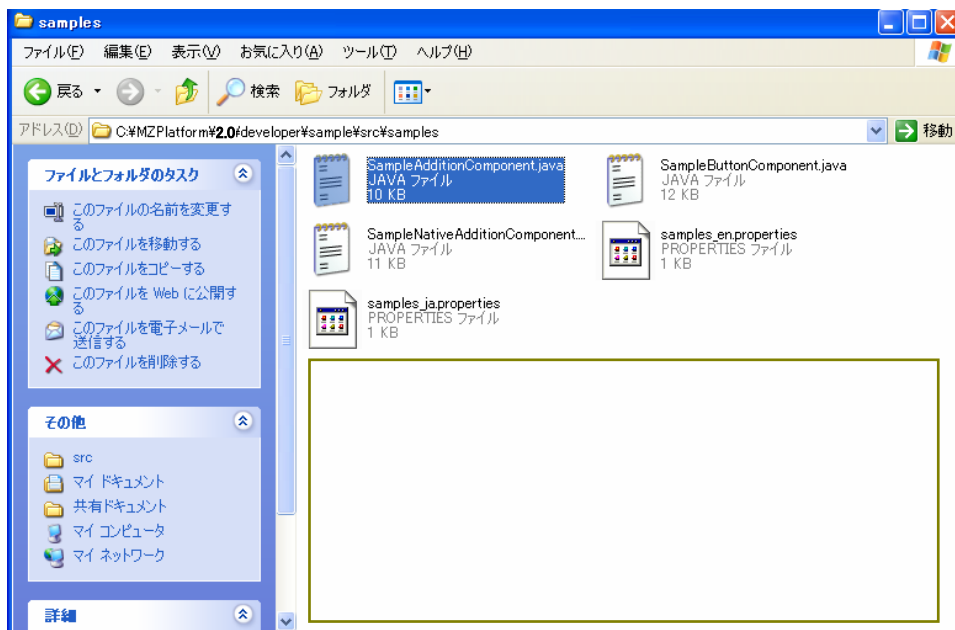
## 手順2 サンプルプログラム名の変更

### 2-1 ファイル名の変更

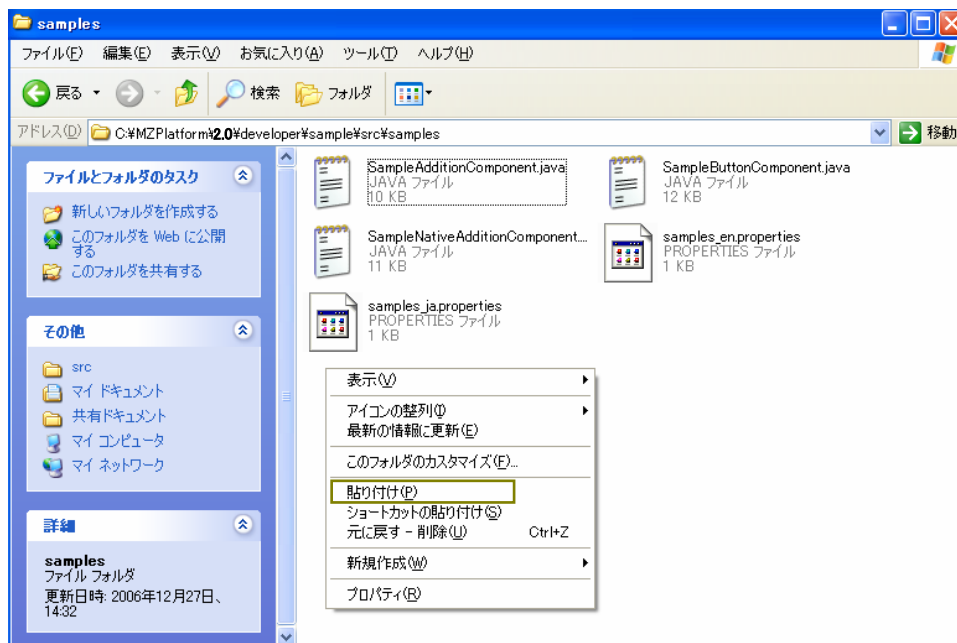
拡張子が「.java」のファイルを1つ選び、アイコン上で右クリックします（例題では、「SampleAdditionComponent.java」を選択）。「コピー(C)」を左クリックします。



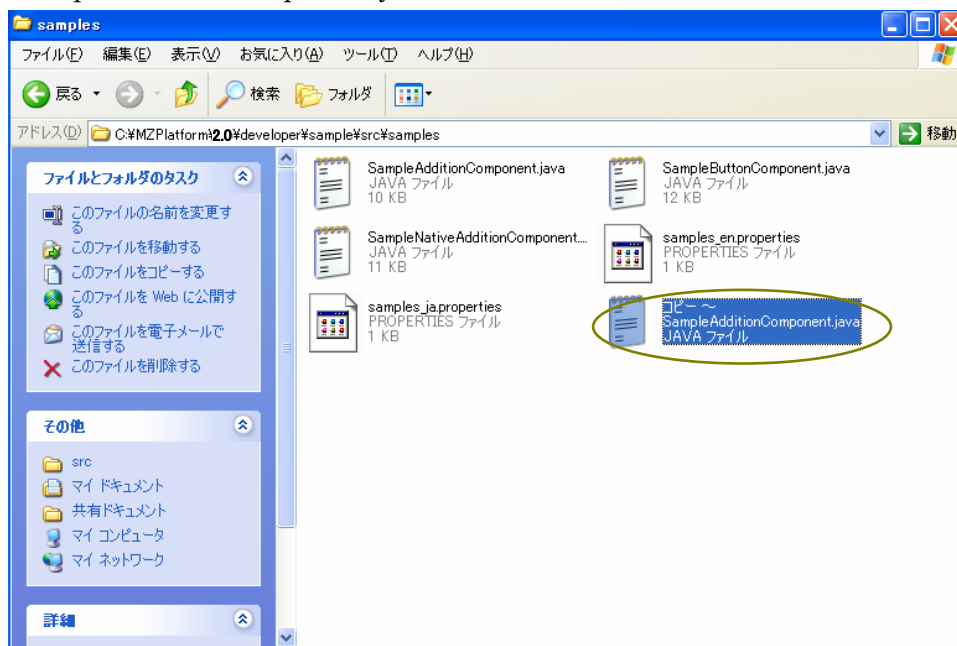
下図の囲み内で右クリックします。



「貼り付け(P)」を左クリックします。

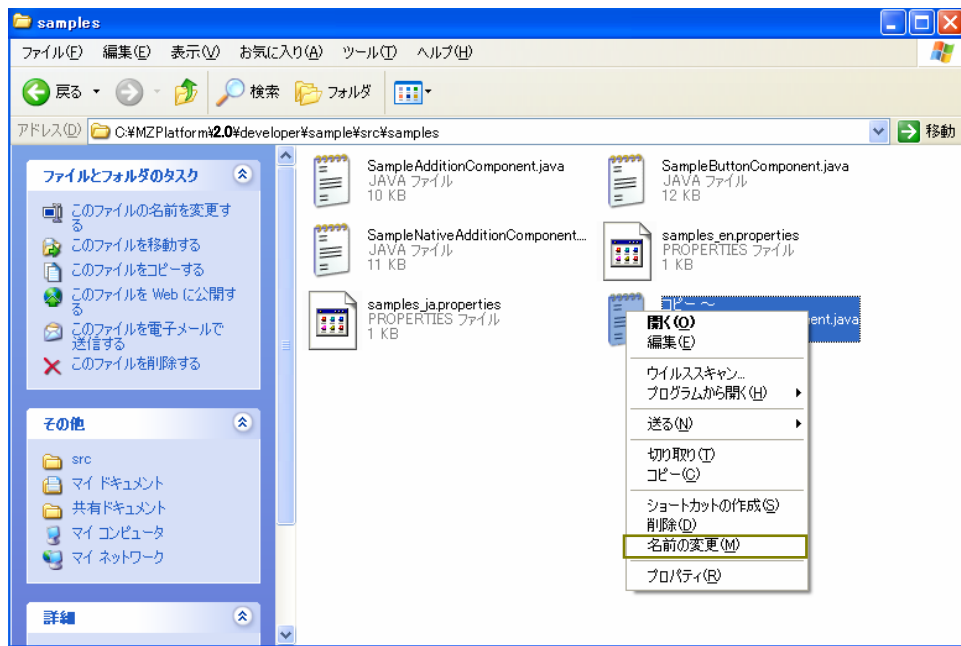


「コピー～SampleAdditionComponent.java」ファイルの追加が確認できます。

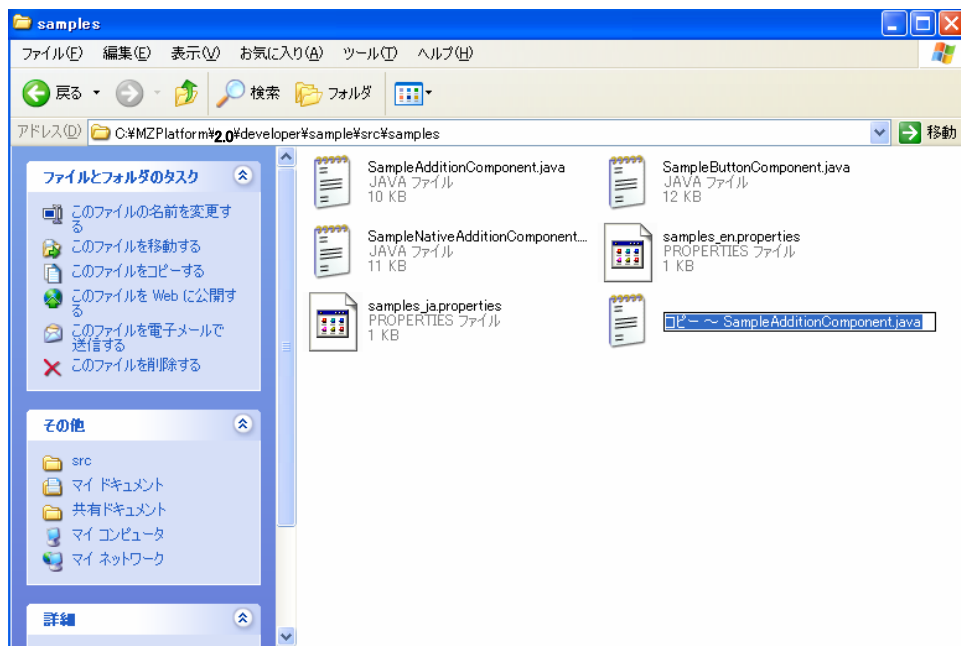




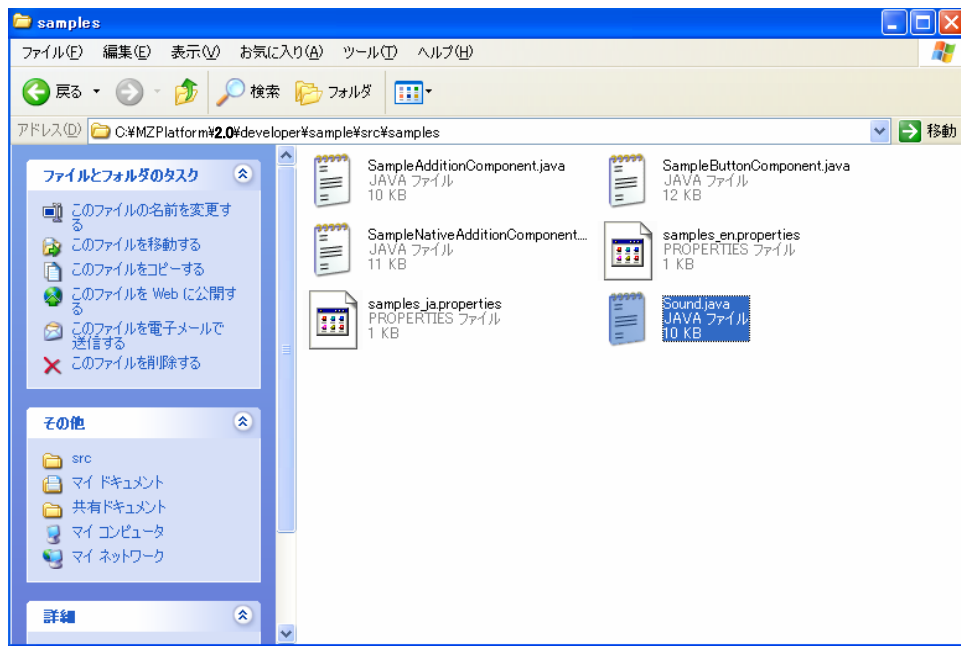
このファイルの名前を変更します。「コピー～SampleAdditionComponent.java」アイコン上で右クリックします。「名前の変更(M)」を左クリックします。



編集できる状態になりました。



例題では、音を鳴らすコンポーネントを作成するので「Sound.java」とします。ファイル名を入力後、[Enter]キーを押して確定させます。

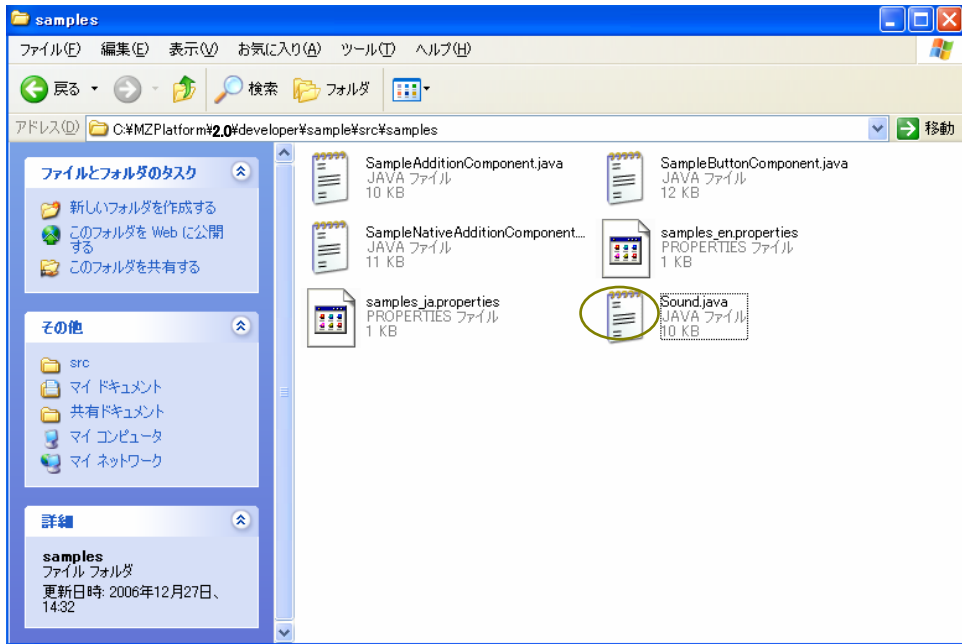


**\*\*補足\*\***

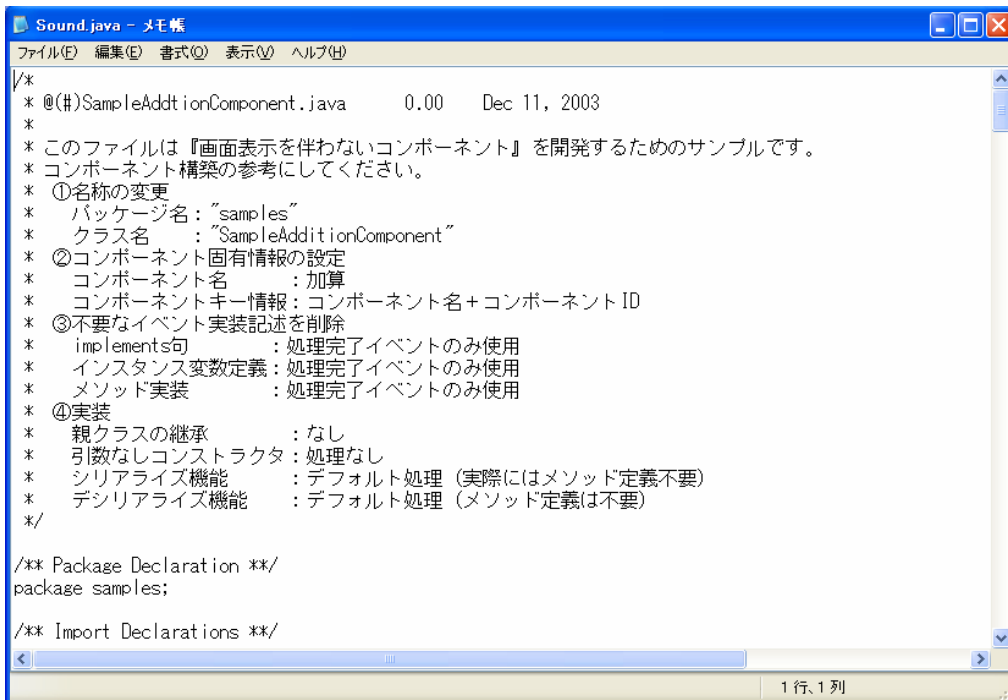
説明では、サンプルファイルを直接編集せずにコピーファイルを別に用意しました。これは元のファイルを残して置くためです（今後参考にする時の事を考えて）。

## 2-2 プログラムコードの編集

「Sound.java」ファイルのアイコンをダブルクリックします。



以下の画面が表示されます。

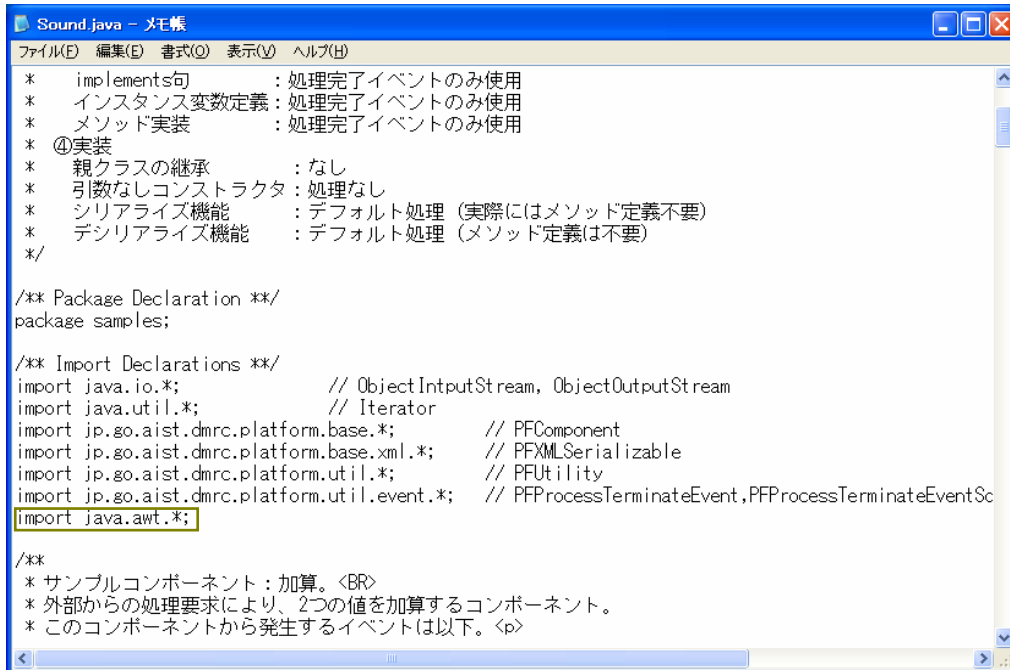


画面を少し下へ移動すると、「import jp.go.aist.dmrplatform.util.event.\*;」の記述があります。この記述の次行に以下の記述を追加します。

```
import java.awt.*;
```

(注1) 入力は半角で行います。

(注2) △部分は半角スペースを入力します。



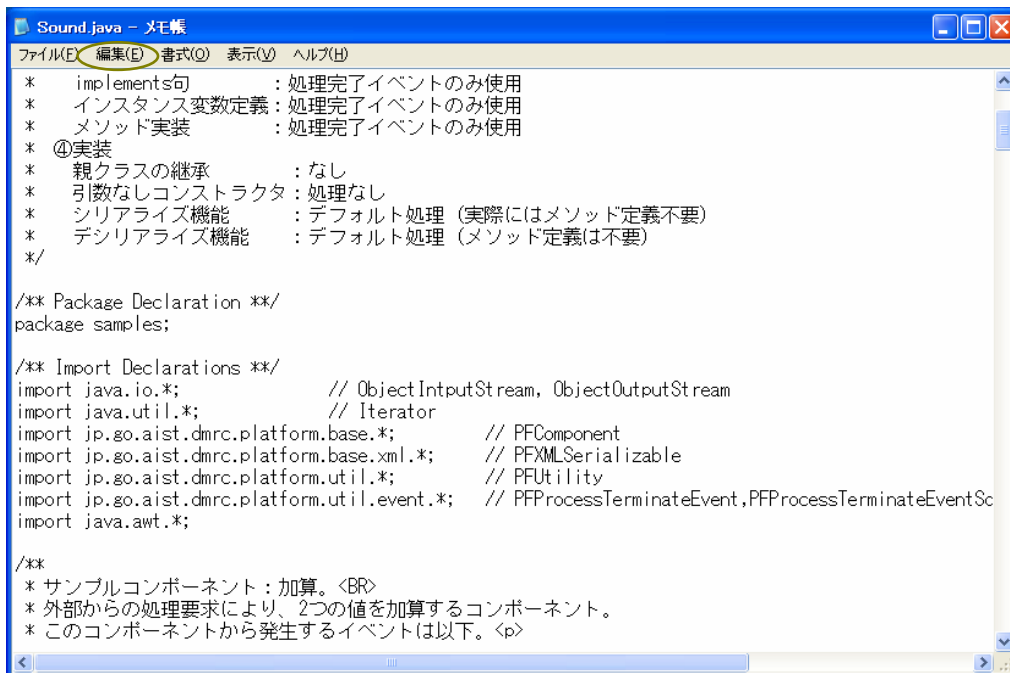
```
Sound.java - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
* implements句      : 処理完了イベントのみ使用
* インスタンス変数定義: 処理完了イベントのみ使用
* メソッド実装      : 処理完了イベントのみ使用
* ④実装
* 親クラスの継承    : なし
* 引数なしコンストラクタ: 処理なし
* シリアライズ機能  : デフォルト処理 (実際にはメソッド定義不要)
* デシリアライズ機能 : デフォルト処理 (メソッド定義は不要)
*/

/** Package Declaration */
package samples;

/** Import Declarations */
import java.io.*;          // ObjectInputStream, ObjectOutputStream
import java.util.*;       // Iterator
import jp.go.aist.dmrplatform.base.*; // PFComponent
import jp.go.aist.dmrplatform.base.xml.*; // PFXMLSerializable
import jp.go.aist.dmrplatform.util.*; // PFUtility
import jp.go.aist.dmrplatform.util.event.*; // PFProcessTerminateEvent, PFProcessTerminateEventSc
import java.awt.*;

/**
 * サンプルコンポーネント：加算。<BR>
 * 外部からの処理要求により、2つの値を加算するコンポーネント。
 * このコンポーネントから発生するイベントは以下。<p>
 */
```

「SampleAdditionComponent」の記述を「Sound」に変更します。変更箇所が複数あるので、[置換]機能を利用します。メニューバーの「編集(E)」を左クリックします。



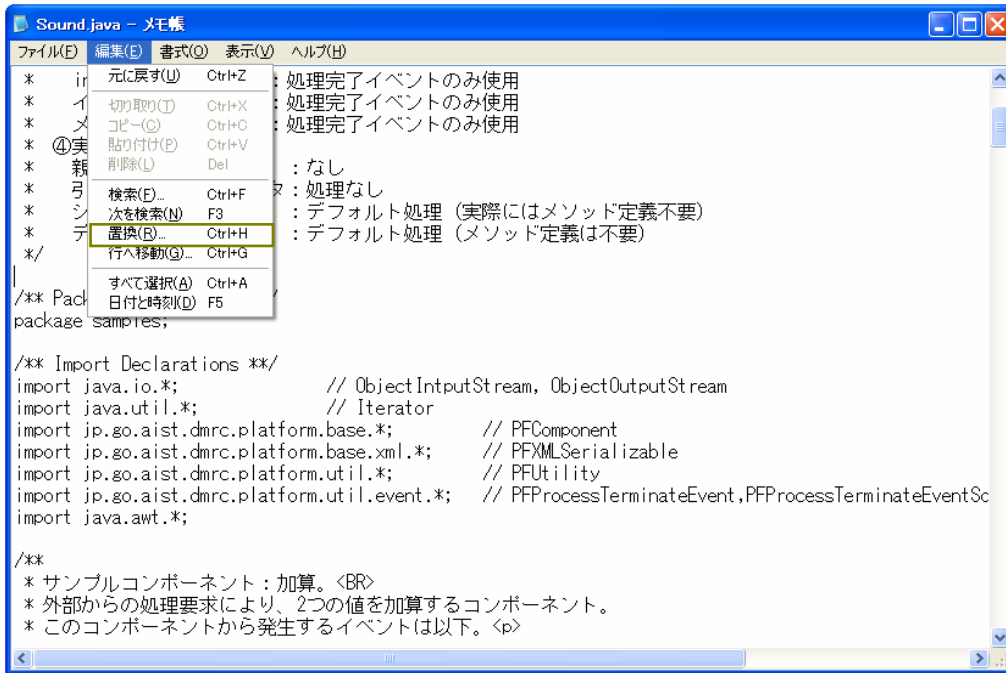
```
Sound.java - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
* implements句      : 処理完了イベントのみ使用
* インスタンス変数定義: 処理完了イベントのみ使用
* メソッド実装      : 処理完了イベントのみ使用
* ④実装
* 親クラスの継承    : なし
* 引数なしコンストラクタ: 処理なし
* シリアライズ機能  : デフォルト処理 (実際にはメソッド定義不要)
* デシリアライズ機能 : デフォルト処理 (メソッド定義は不要)
*/

/** Package Declaration */
package samples;

/** Import Declarations */
import java.io.*;          // ObjectInputStream, ObjectOutputStream
import java.util.*;       // Iterator
import jp.go.aist.dmrplatform.base.*; // PFComponent
import jp.go.aist.dmrplatform.base.xml.*; // PFXMLSerializable
import jp.go.aist.dmrplatform.util.*; // PFUtility
import jp.go.aist.dmrplatform.util.event.*; // PFProcessTerminateEvent, PFProcessTerminateEventSc
import java.awt.*;

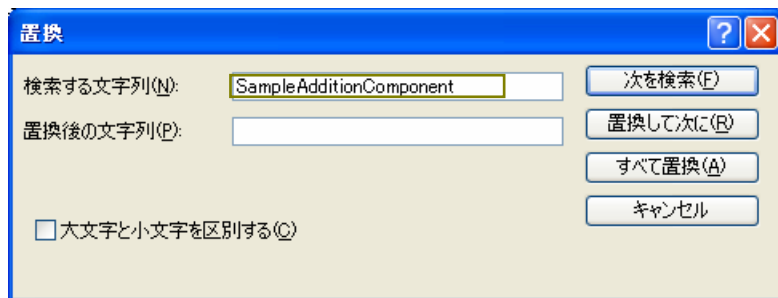
/**
 * サンプルコンポーネント：加算。<BR>
 * 外部からの処理要求により、2つの値を加算するコンポーネント。
 * このコンポーネントから発生するイベントは以下。<p>
 */
```

一覧から「置換(R)...」を左クリックします。



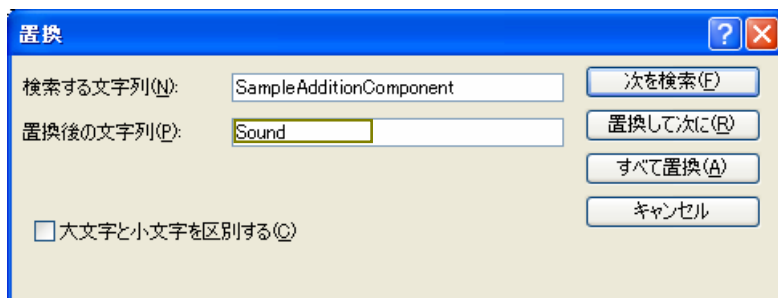
「置換」画面が表示されます。「検索する文字列(N):」右隣のボックスに以下の記述を行います。

SampleAdditionComponent  
(注) 入力は半角で行います。

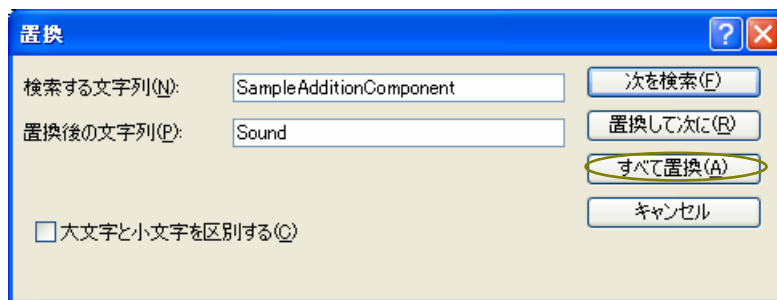


「置換後の文字列(P):」の右隣のボックスに以下の記述を行います。

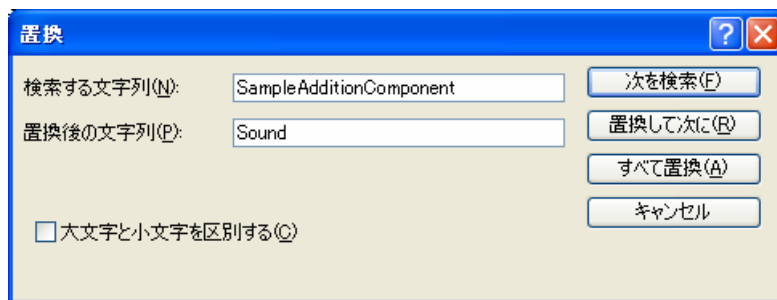
Sound  
(注) 入力は半角で行います。



「すべて置換(A)」を左クリックします。

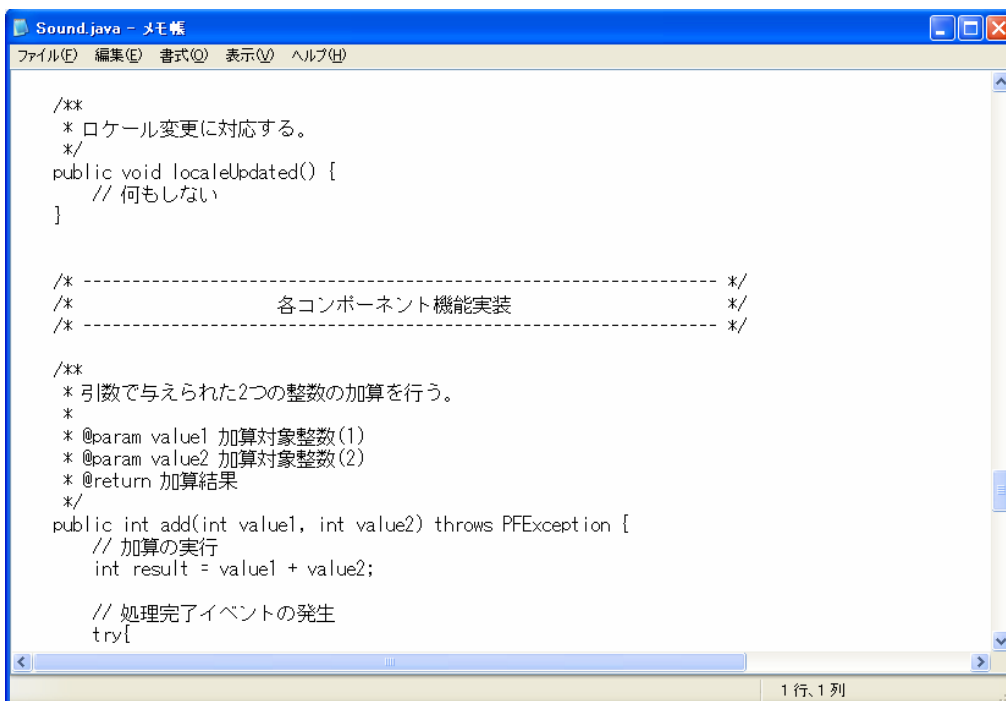


「×」ボタンを左クリックし、「置換」画面を閉じます。



画面下へ約 2/3 移動したところに、以下の記述があります。

```
Public void localeUpdated{
//何もしない
}
```



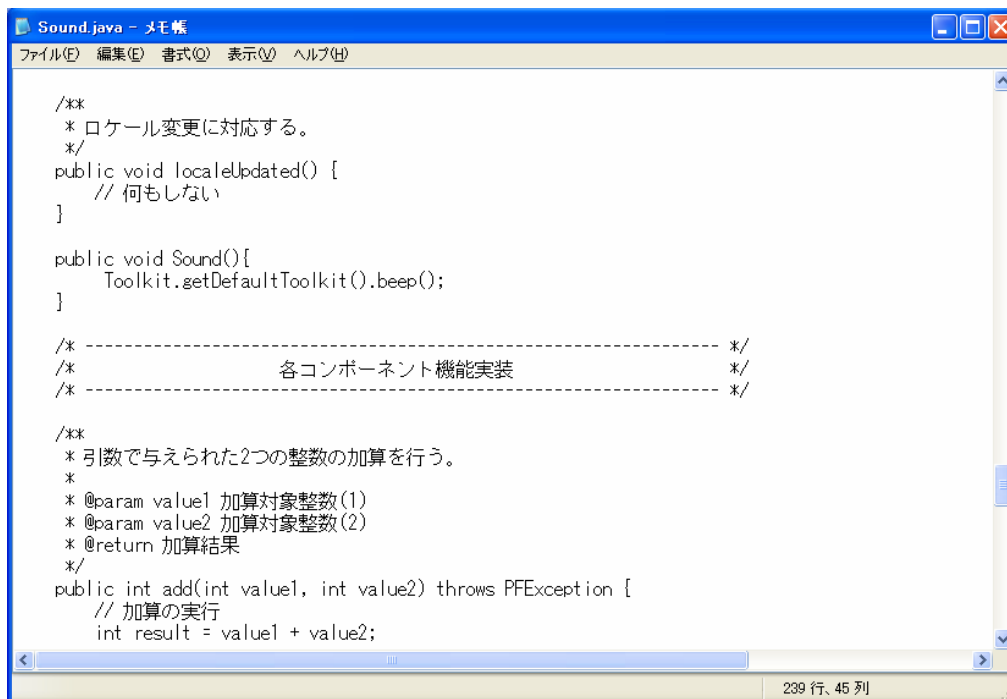
この記述の下に1行改行を入れて、以下の記述を追加します。

```
public△void△Sound(){  
    Toolkit.getDefaultToolkit().beep();  
}
```

(注1) 入力は半角で行います。

(注2) △部分は半角スペースを入力します。

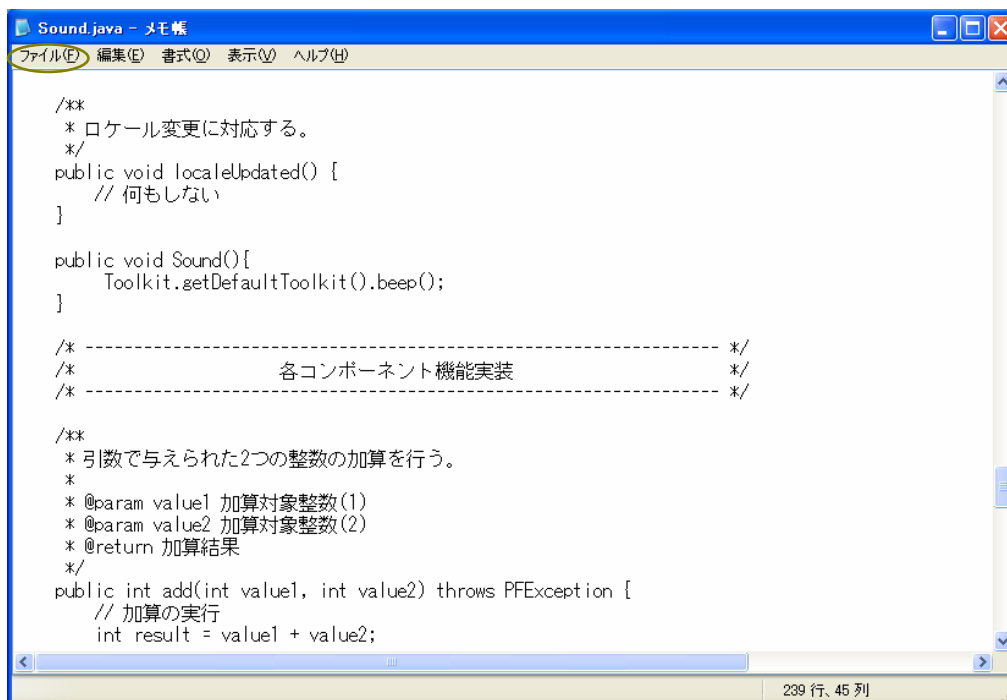
(注3) 他のコンポーネントを作成する場合には、他のコードを記述します。



```
Sound.java - メモ帳  
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)  
  
/**  
 * ロケール変更に対応する。  
 */  
public void localeUpdated() {  
    // 何もしない  
}  
  
public void Sound(){  
    Toolkit.getDefaultToolkit().beep();  
}  
  
/* ----- */  
/*                各コンポーネント機能実装                */  
/* ----- */  
  
/**  
 * 引数で与えられた2つの整数の加算を行う。  
 *  
 * @param value1 加算対象整数(1)  
 * @param value2 加算対象整数(2)  
 * @return 加算結果  
 */  
public int add(int value1, int value2) throws PFXException {  
    // 加算の実行  
    int result = value1 + value2;  
}
```

239行, 45列

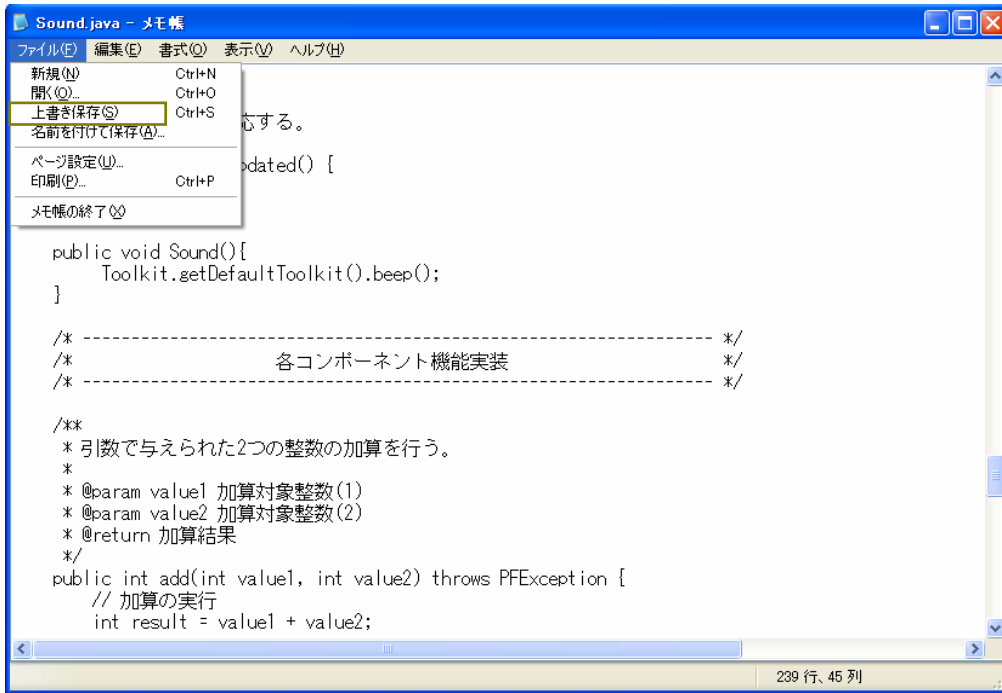
ここで行った設定を保存します。メニューバーの「ファイル(F)」を左クリックします。



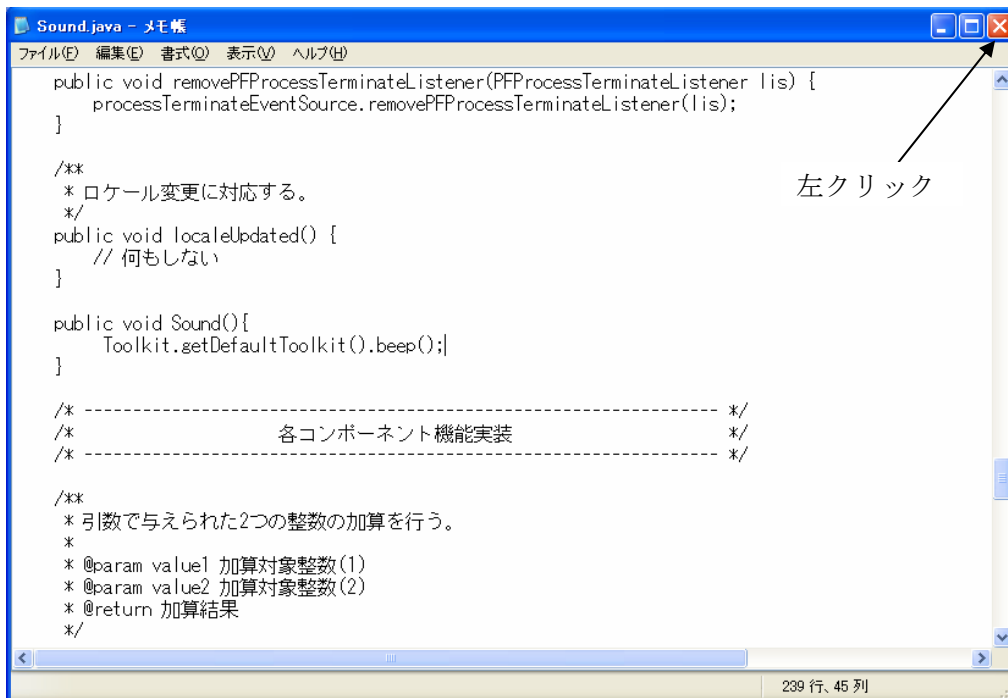
```
Sound.java - メモ帳  
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)  
  
/**  
 * ロケール変更に対応する。  
 */  
public void localeUpdated() {  
    // 何もしない  
}  
  
public void Sound(){  
    Toolkit.getDefaultToolkit().beep();  
}  
  
/* ----- */  
/*                各コンポーネント機能実装                */  
/* ----- */  
  
/**  
 * 引数で与えられた2つの整数の加算を行う。  
 *  
 * @param value1 加算対象整数(1)  
 * @param value2 加算対象整数(2)  
 * @return 加算結果  
 */  
public int add(int value1, int value2) throws PFXException {  
    // 加算の実行  
    int result = value1 + value2;  
}
```

239行, 45列

一覧から[上書き保存(S)]を左クリックします。



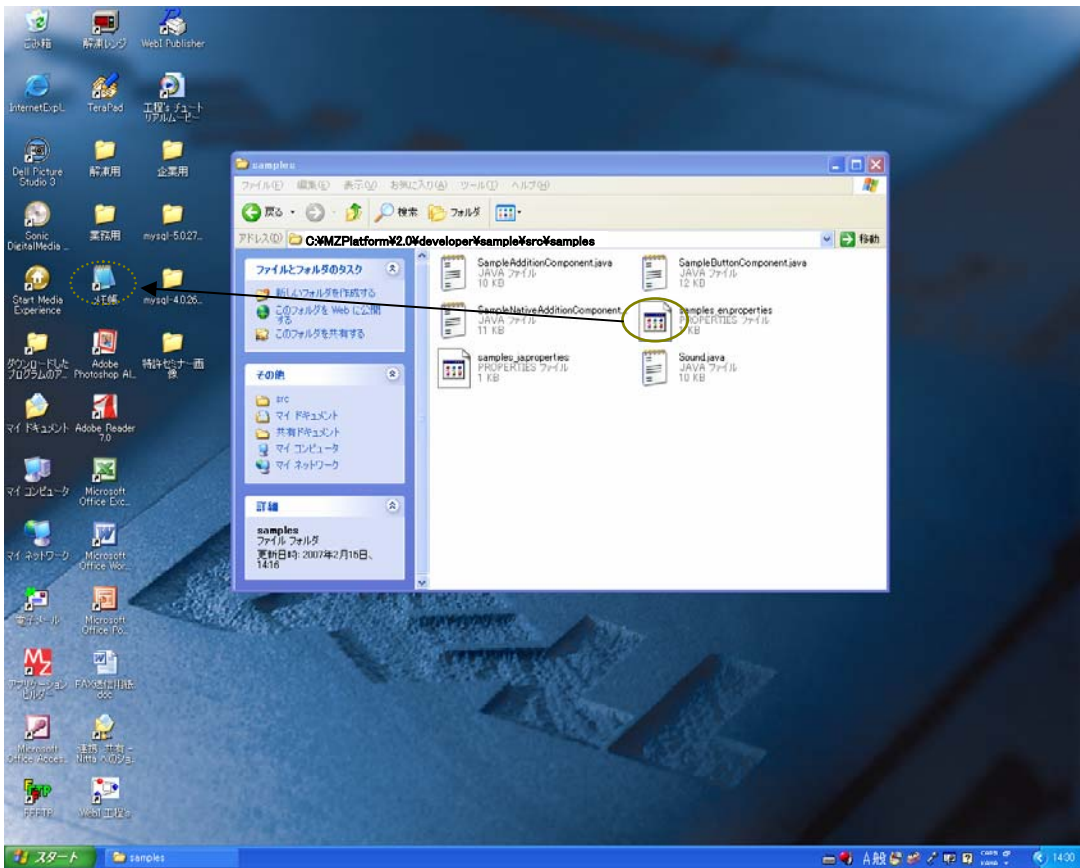
「×」ボタンを左クリックし、「Sound.java」ファイルを閉じます。



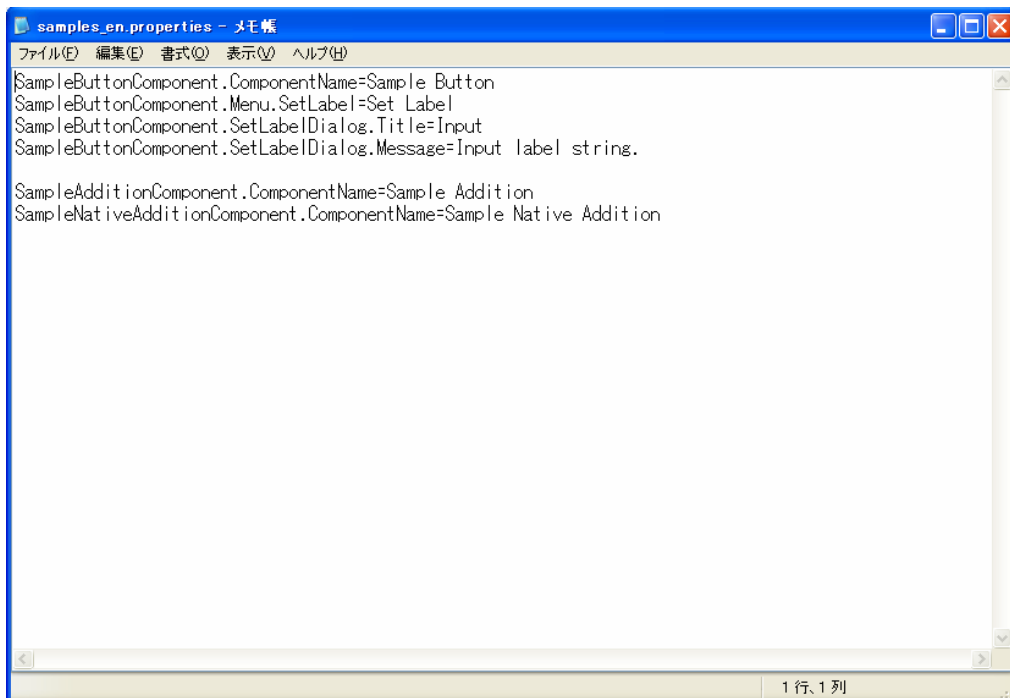


### 2-3 「sample\_en.properties」ファイルの修正

「samples\_en.properties」ファイルのアイコンを、「メモ帳」アイコン上にドラッグ&ドロップします。「メモ帳」アイコンがデスクトップ上にない場合には、「スタート」→「すべてのプログラム」→「アクセサリ」→「メモ帳」とたどり、表示されたメモ帳のウィンドウ上にドラッグ&ドロップします。



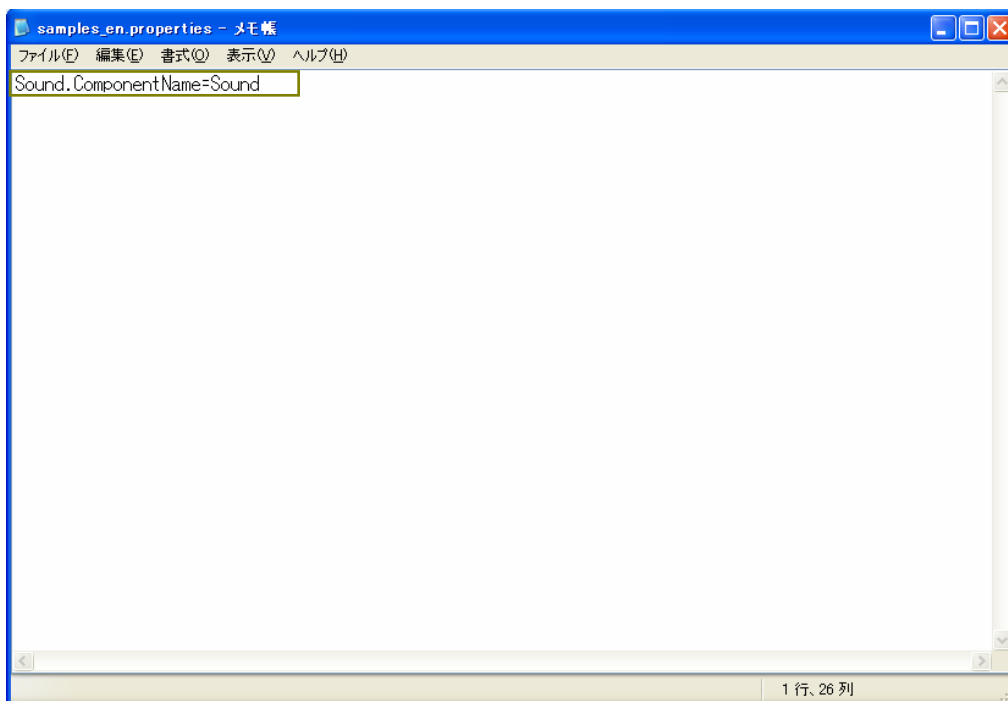
以下の画面が表示されます。



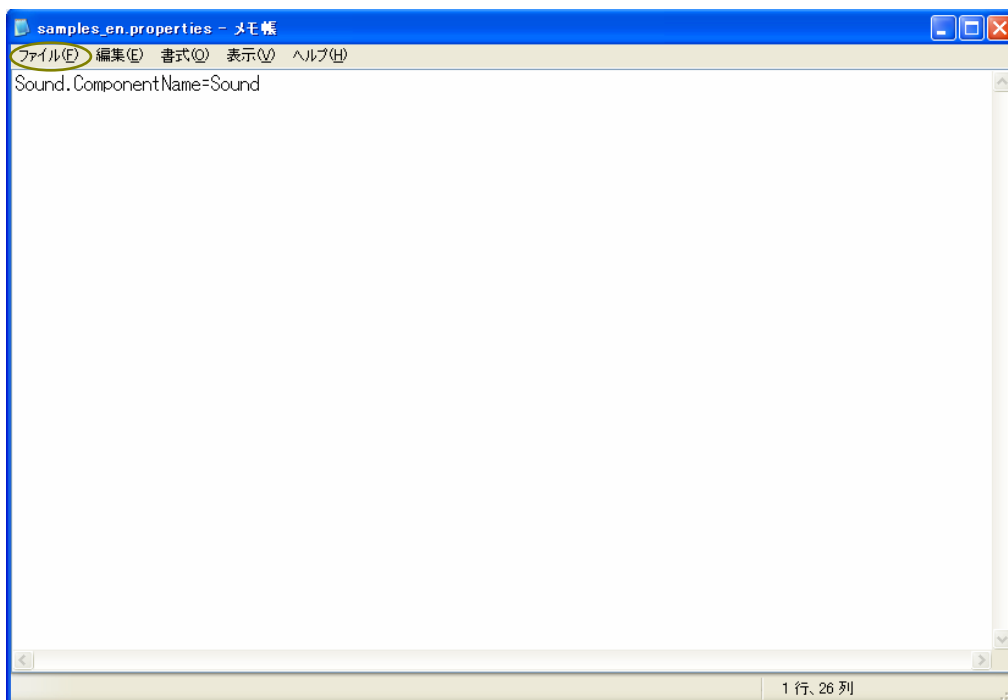
既存のプログラムコードを消去して、以下のコードを記述します。

```
Sound.ComponentName=Sound
```

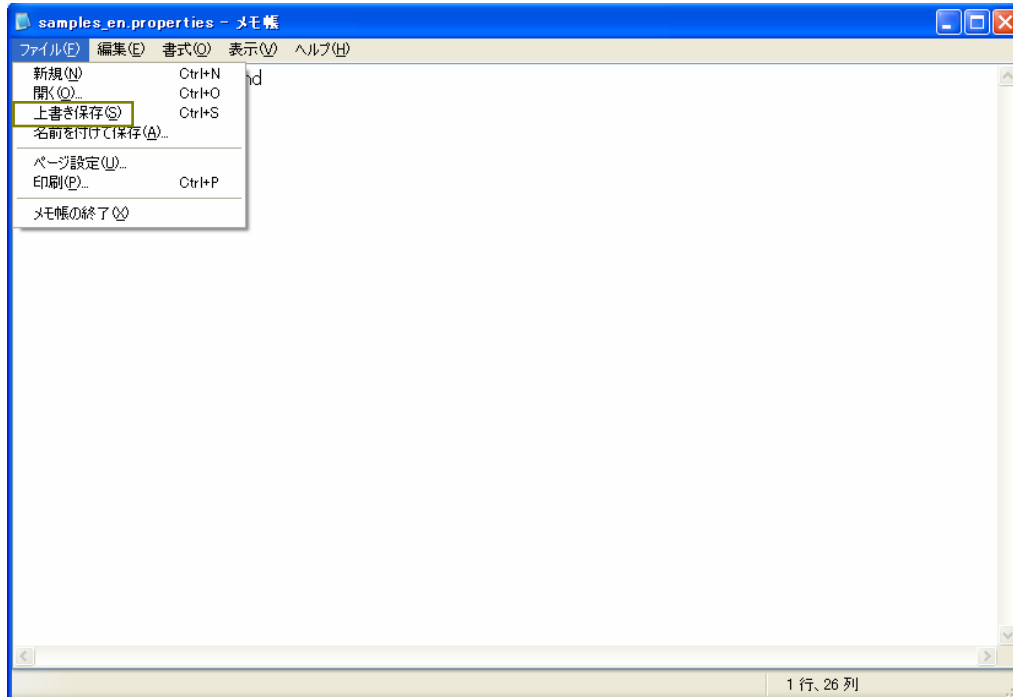
(注) 入力は半角で行います。



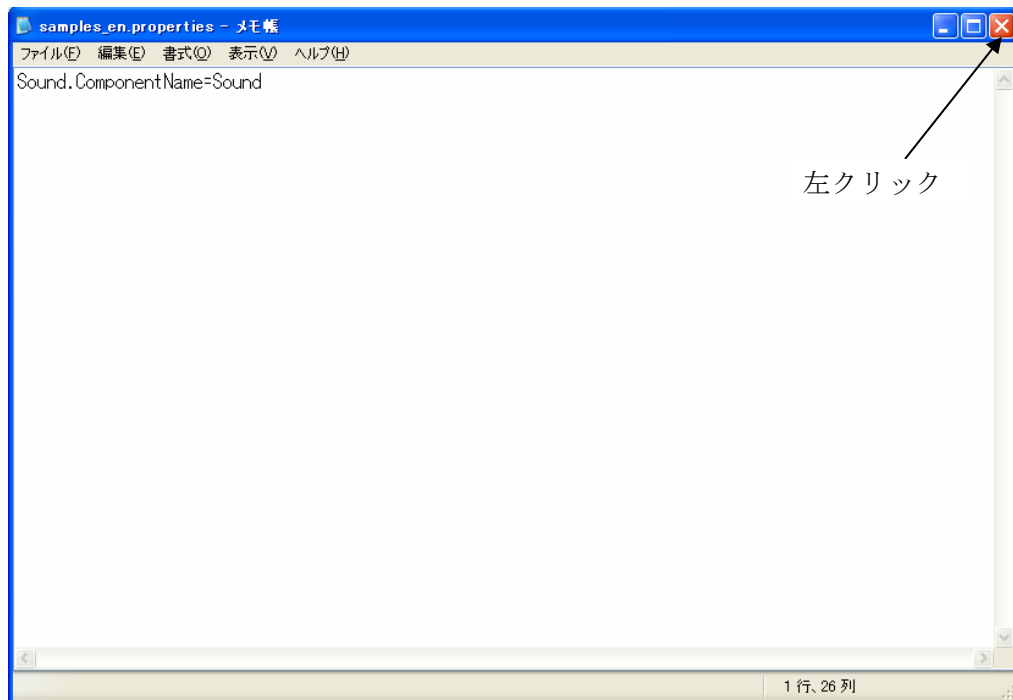
ここで行った設定を保存します。メニューバーの「ファイル(F)」を左クリックします。



一覧から「上書き保存(S)」を左クリックします。

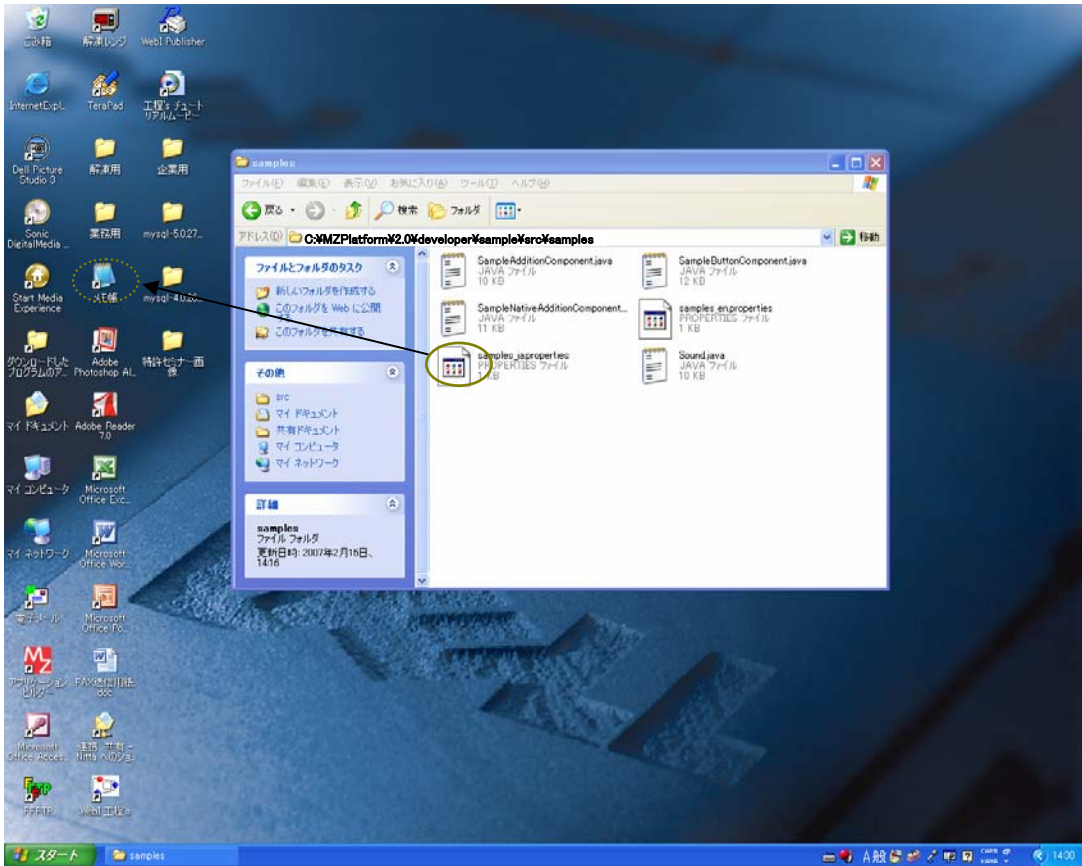


「×」ボタンを左クリックし、「samples\_en.properties」ファイルを閉じます。

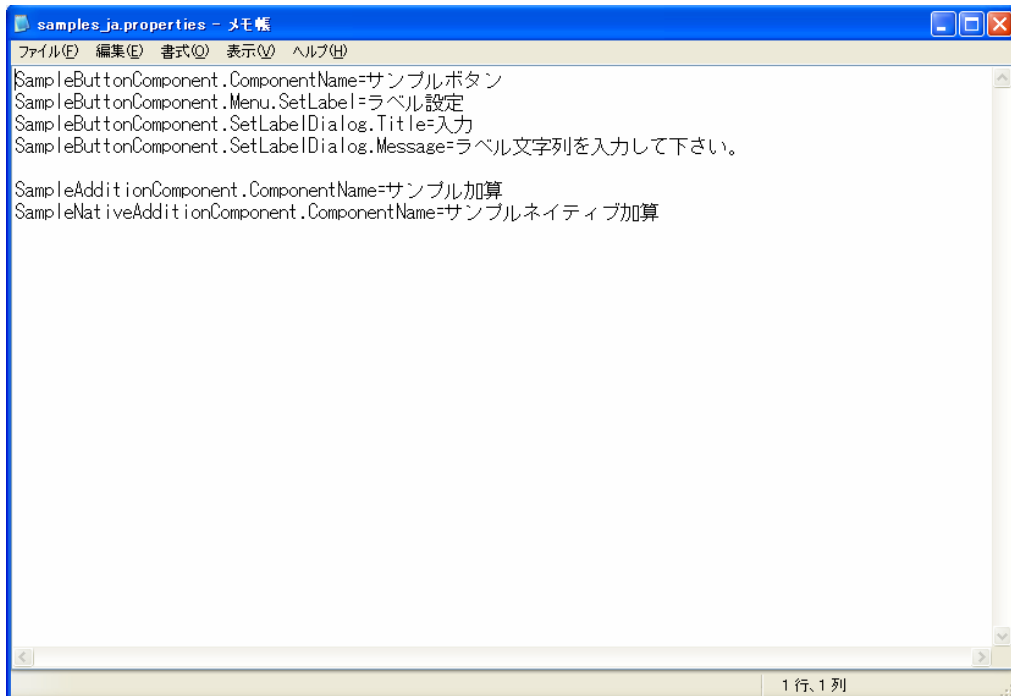


## 2-4 「samples\_ja.properties」ファイルの修正

「samples\_ja.properties」ファイルのアイコンを、「メモ帳」アイコン上にドラッグ&ドロップします。「メモ帳」アイコンがデスクトップ上にはない場合には、「スタート」→「すべてのプログラム」→「アクセサリ」→「メモ帳」とどり、表示されたメモ帳のウィンドウ上にドラッグ&ドロップします。



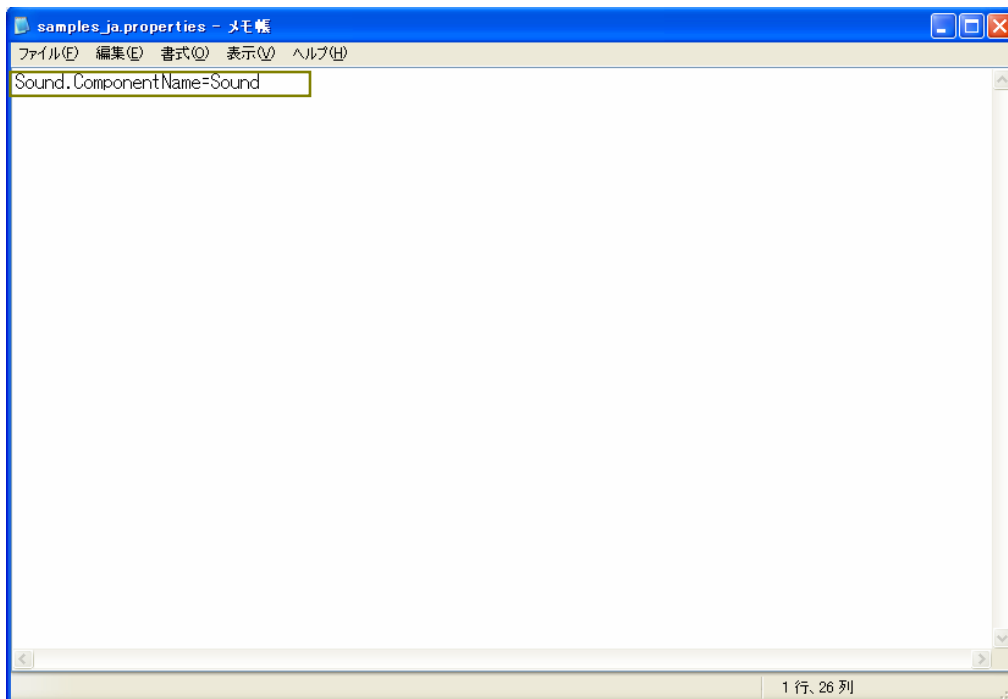
以下の画面が表示されます。



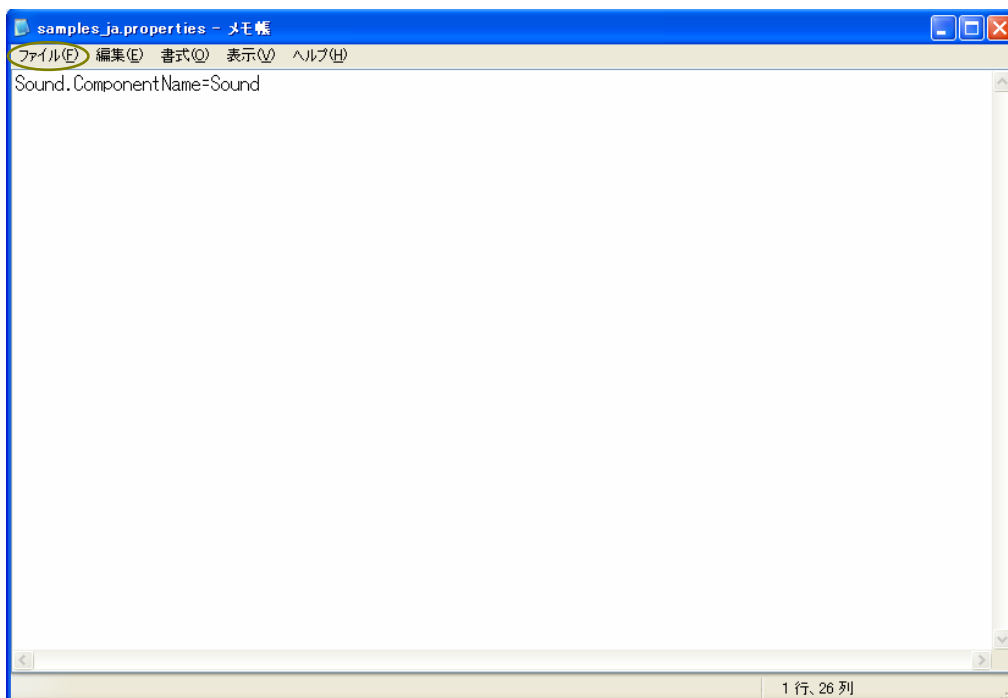
既存のプログラムを消去して、以下のコードを記述します。

```
Sound.ComponentName=Sound
```

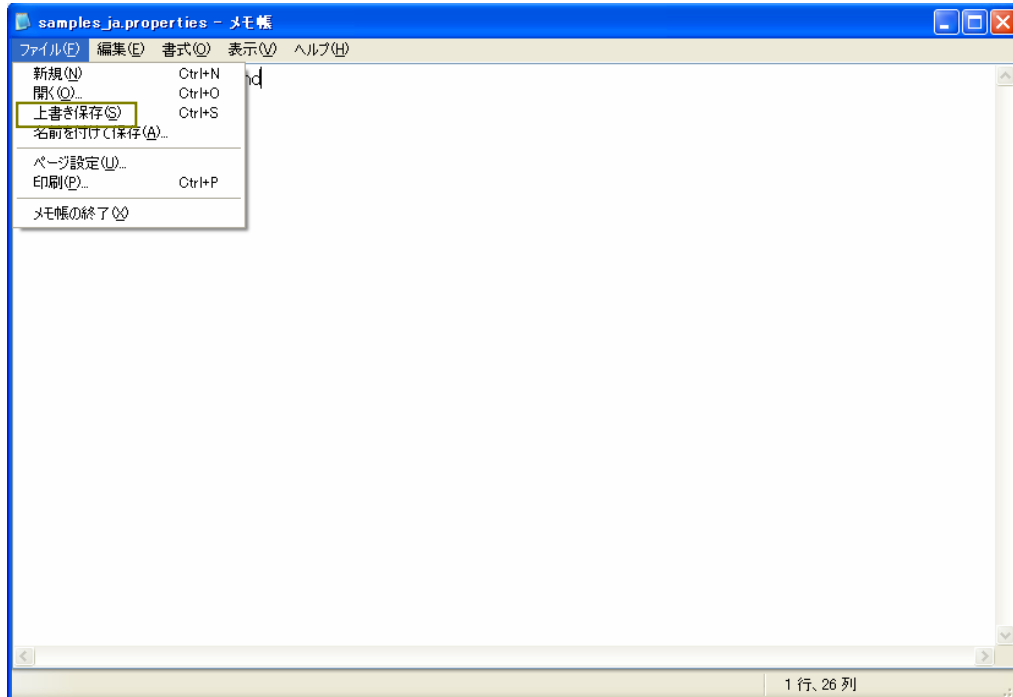
(注) 入力は半角で行います。



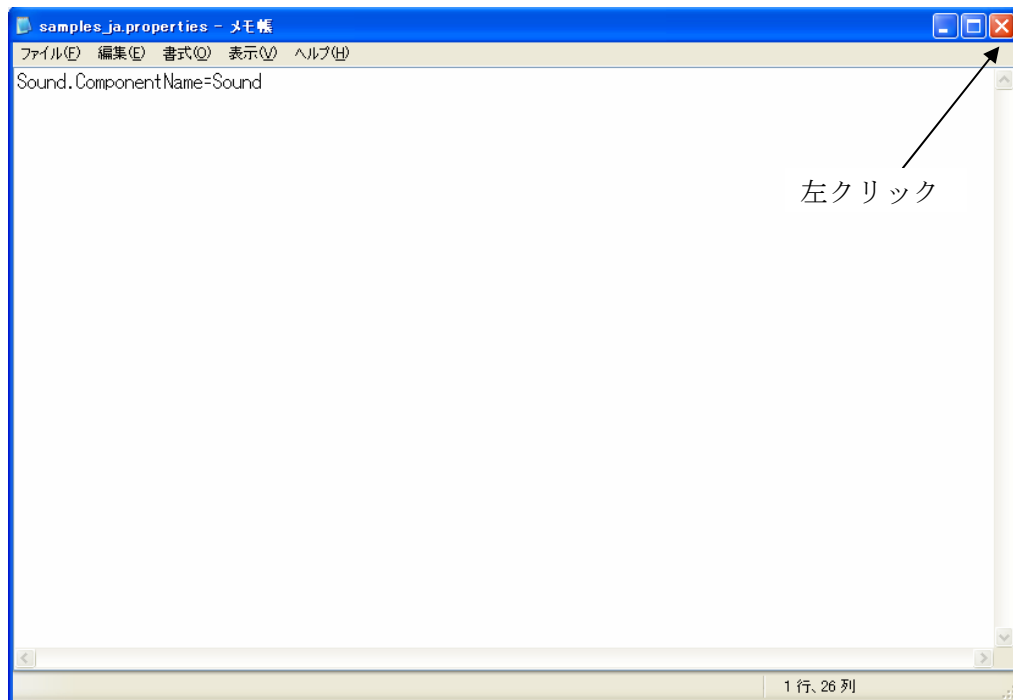
ここで行った設定を保存します。メニューバーの「ファイル(F)」を左クリックします。



一覧から「上書き保存(S)」を左クリックします。



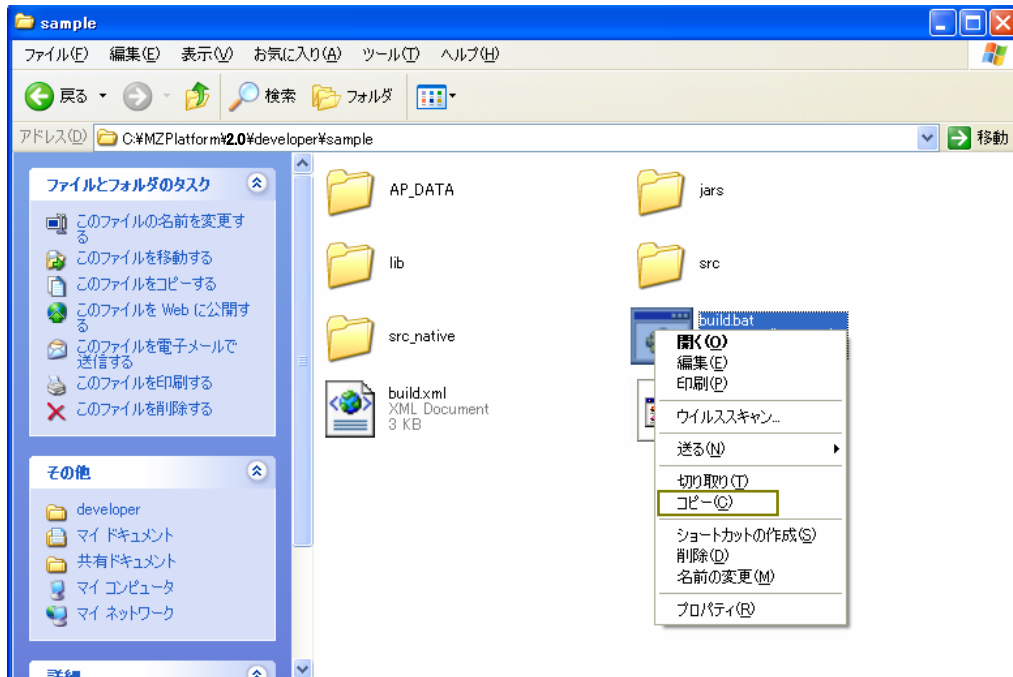
「×」ボタンを左クリックし、「samples\_ja.properties」ファイルを閉じます。



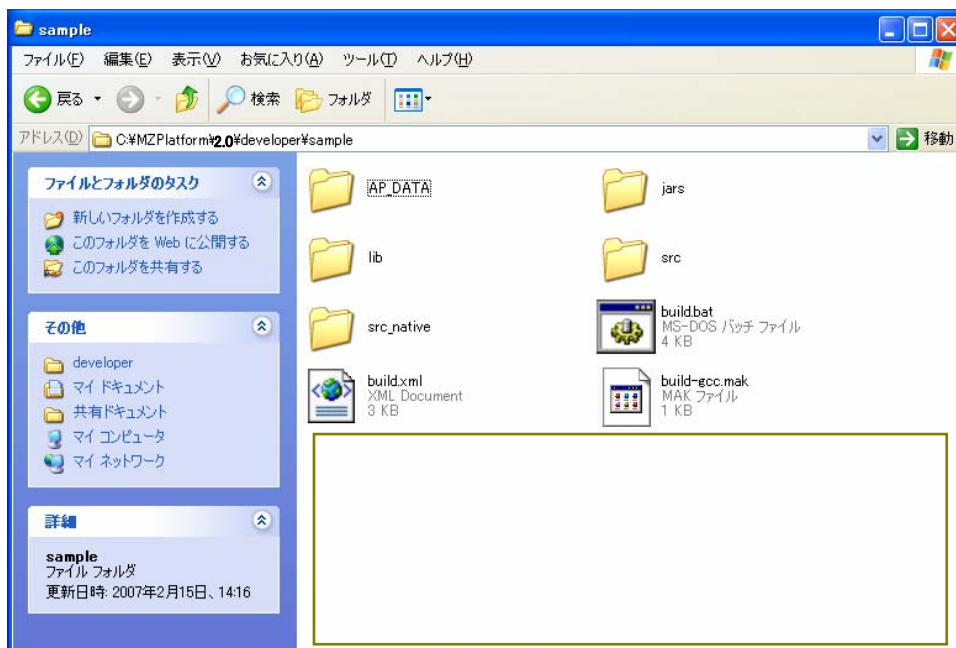
### 手順3 「build.bat」ファイルの編集

#### 3-1 ファイルのコピー

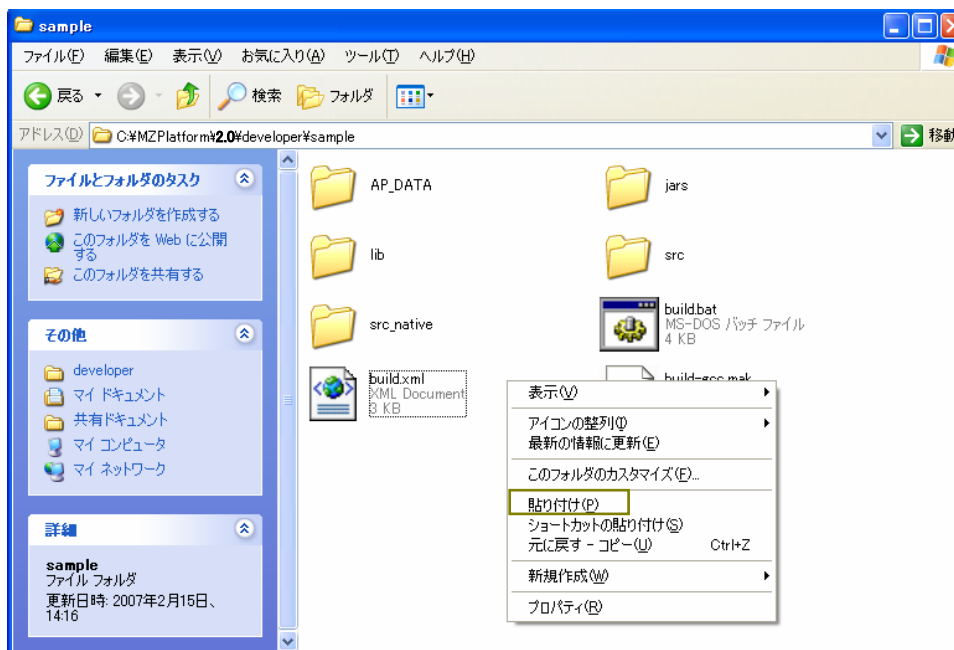
「スタート」→「マイコンピュータ」→「ローカルディスク(C:)」→「MZPlatform」→「2.0」→「developer」→「sample」とたどると、「build.bat」ファイルがあります。「build.bat」ファイルのアイコン上で右クリックし、「コピー(C)」を左クリックします。



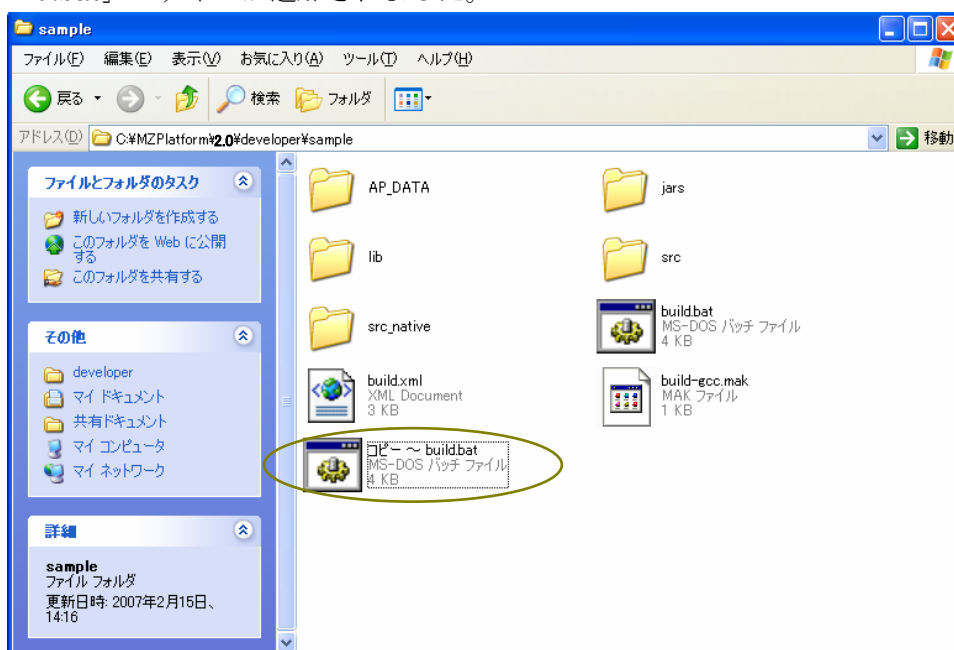
下図の囲み内で右クリックします。



「貼り付け(P)」を左クリックします。

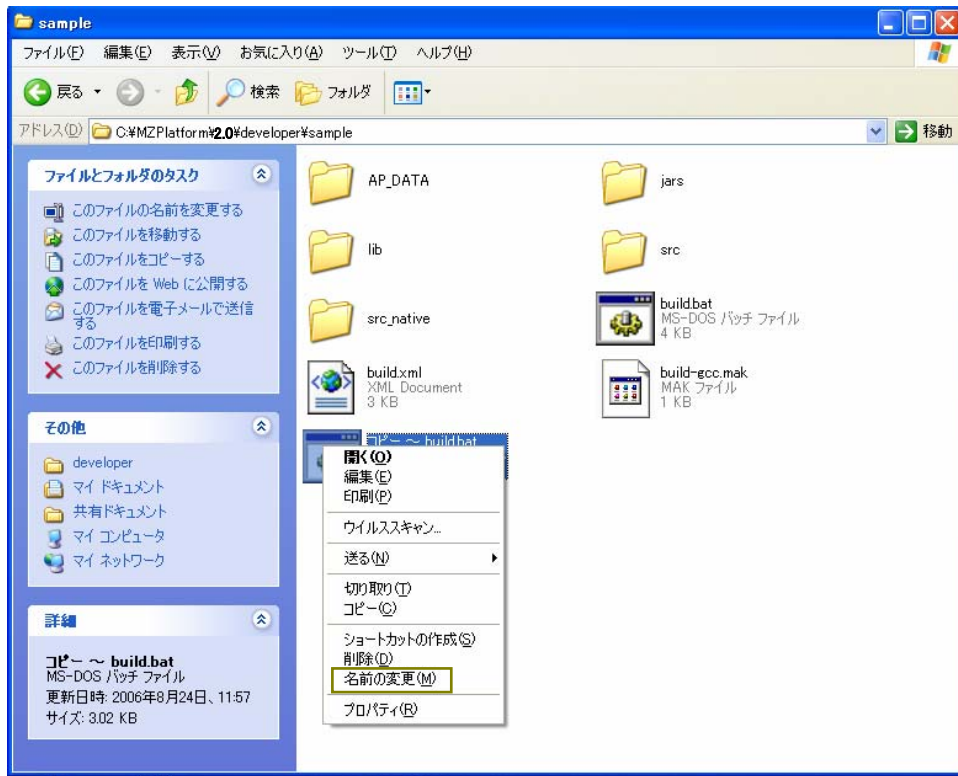


「コピー～build.bat」ファイルが追加されました。

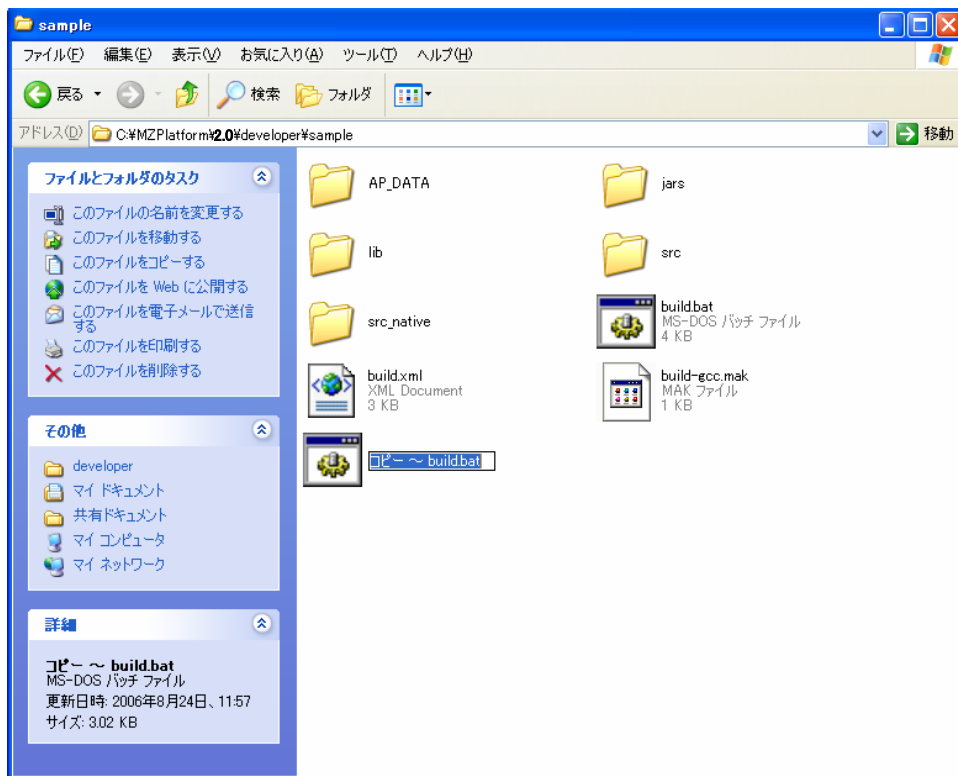




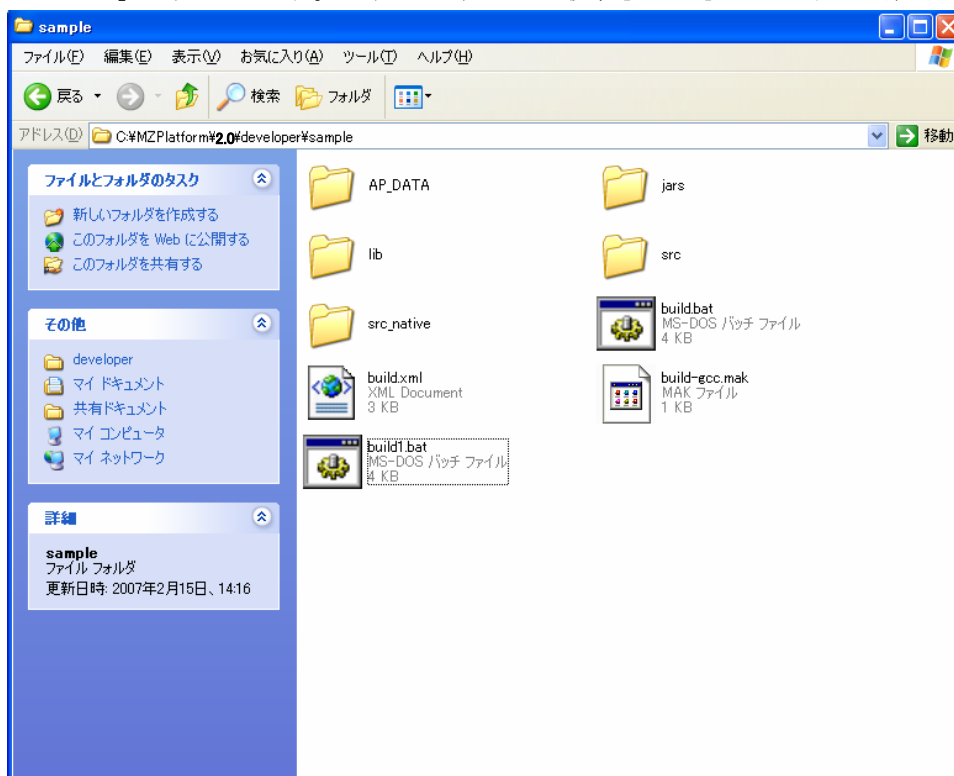
このファイルの名前を変更します。「コピー～build.bat」ファイルのアイコン上で右クリックします。「名前の変更(M)」を左クリックします。



編集できる状態になりました。



例題では「build1.bat」と変更します。ファイル名を入力後、[Enter]キーを押して確定させます。

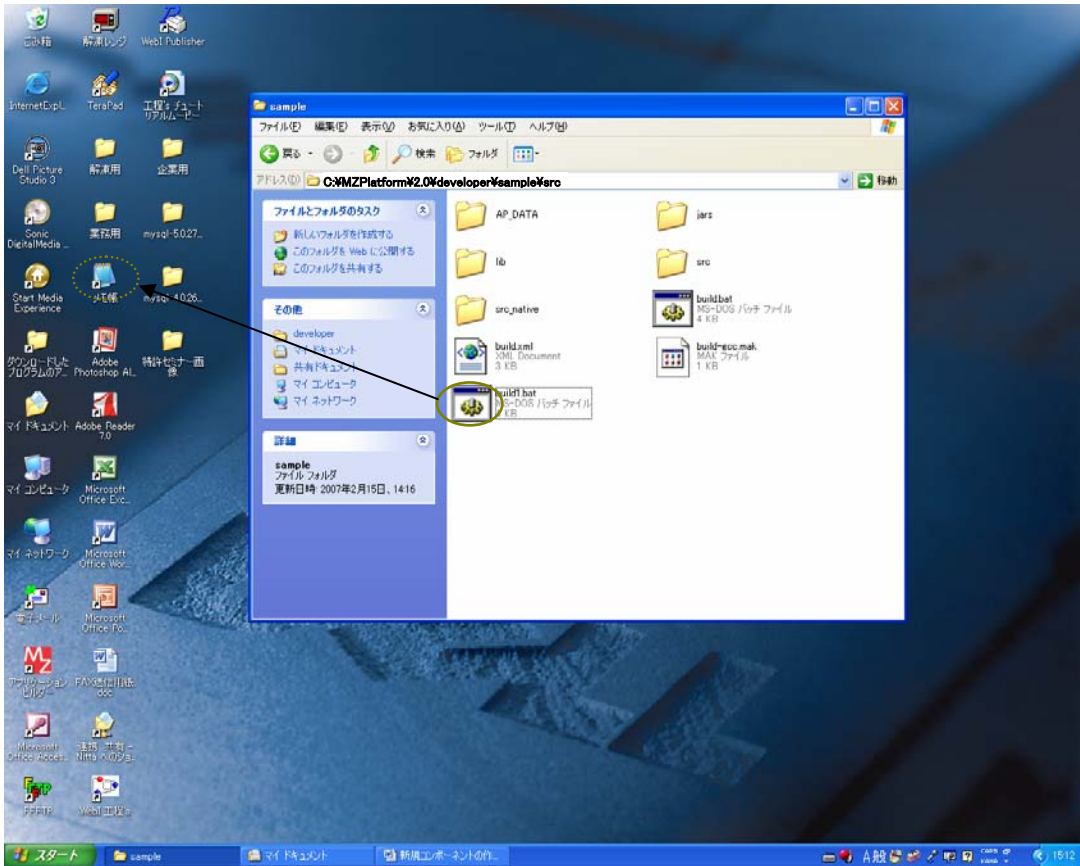


**\*\*補足\*\***

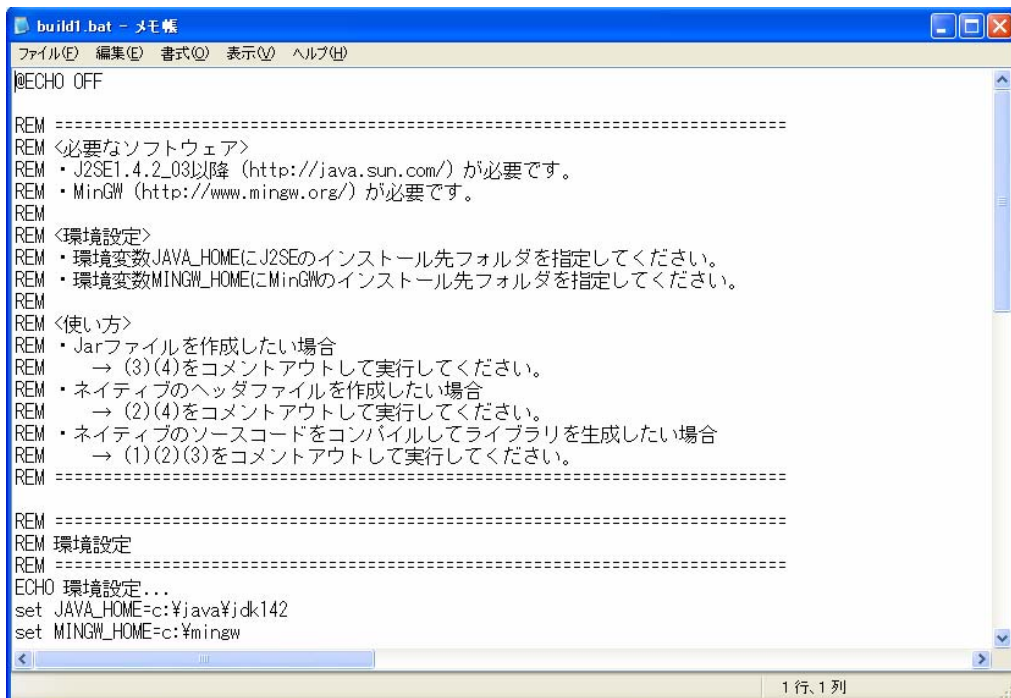
説明では、サンプルファイルを直接編集せずにコピーファイルを別に用意しました。  
これは元のファイルを残しておくためです（今後参考にする時の事を考えて）。

### 3-2 バッチファイルのプログラム編集

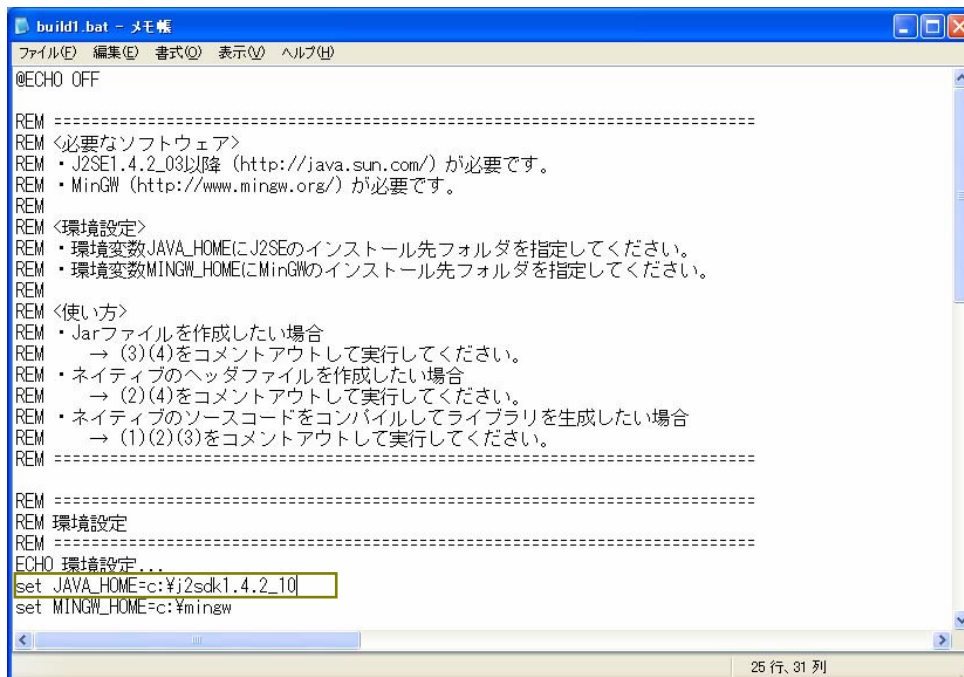
「build1.bat」ファイルのアイコンを、「メモ帳」アイコン上にドラッグ&ドロップします。「メモ帳」アイコンがデスクトップ上にはない場合には、「スタート」→「すべてのプログラム」→「アクセサリ」→「メモ帳」とどり、表示されたメモ帳のウィンドウ上にドラッグ&ドロップします。



以下の画面が表示されます。



「ECHO 環境設定...」の次行に、「set JAVA\_HOME=c:¥java¥jdk142」の記述があります。青色文字の部分を、Java 開発環境 (JDK) のインストール先フォルダに変更します (例えば、set JAVA\_HOME=c:¥j2sdk1.4.2\_10 と記述)。JDK は、別途入手し、インストールする必要があります<sup>1</sup>。



```
build1.bat - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
@ECHO OFF

REM =====
REM <必要なソフトウェア>
REM ・ J2SE1.4.2_03以降 (http://java.sun.com/) が必要です。
REM ・ MinGW (http://www.mingw.org/) が必要です。
REM
REM <環境設定>
REM ・ 環境変数 JAVA_HOME に J2SE のインストール先フォルダを指定してください。
REM ・ 環境変数 MINGW_HOME に MinGW のインストール先フォルダを指定してください。
REM
REM <使い方>
REM ・ Jar ファイルを作成したい場合
REM   → (3)(4) をコメントアウトして実行してください。
REM ・ ネイティブのヘッダファイルを作成したい場合
REM   → (2)(4) をコメントアウトして実行してください。
REM ・ ネイティブのソースコードをコンパイルしてライブラリを生成したい場合
REM   → (1)(2)(3) をコメントアウトして実行してください。
REM =====

REM =====
REM 環境設定
REM =====
ECHO 環境設定...
set JAVA_HOME=c:¥j2sdk1.4.2_10
set MINGW_HOME=c:¥mingw

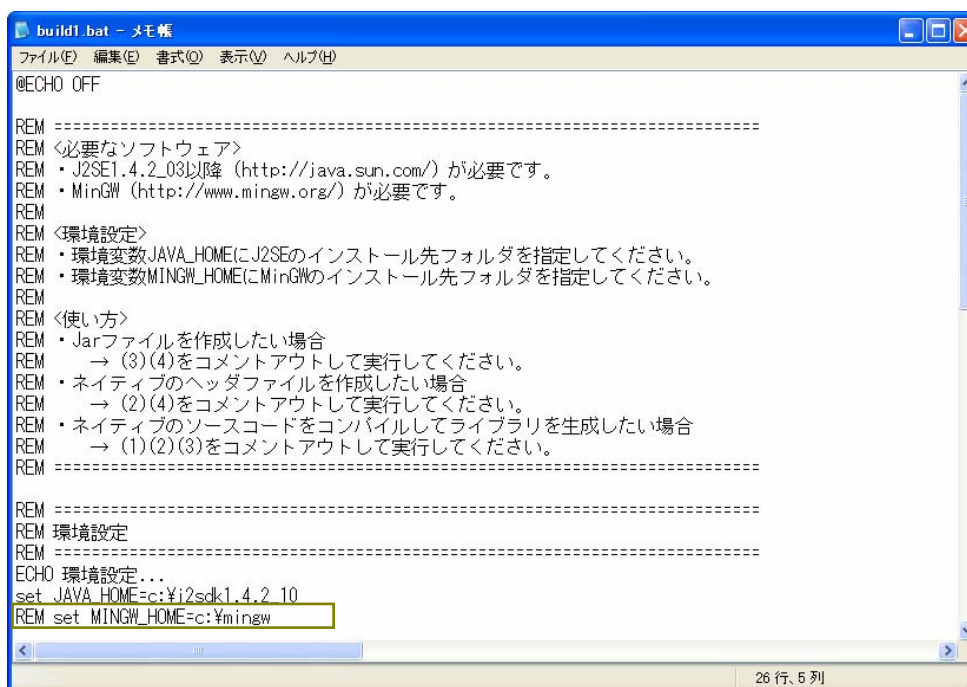
25 行, 31 列
```

次行に「set MINGW\_HOME=c:¥mingw」の記述がありますが、この命令文は実行しないのでコメント (注釈) 文にします。以下の青色部分を追加します。

```
REM△set MINGW_HOME=c:¥mingw
```

(注 1) 入力は半角で行います。

(注 2) △部分は半角スペースを入力します。



```
build1.bat - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
@ECHO OFF

REM =====
REM <必要なソフトウェア>
REM ・ J2SE1.4.2_03以降 (http://java.sun.com/) が必要です。
REM ・ MinGW (http://www.mingw.org/) が必要です。
REM
REM <環境設定>
REM ・ 環境変数 JAVA_HOME に J2SE のインストール先フォルダを指定してください。
REM ・ 環境変数 MINGW_HOME に MinGW のインストール先フォルダを指定してください。
REM
REM <使い方>
REM ・ Jar ファイルを作成したい場合
REM   → (3)(4) をコメントアウトして実行してください。
REM ・ ネイティブのヘッダファイルを作成したい場合
REM   → (2)(4) をコメントアウトして実行してください。
REM ・ ネイティブのソースコードをコンパイルしてライブラリを生成したい場合
REM   → (1)(2)(3) をコメントアウトして実行してください。
REM =====

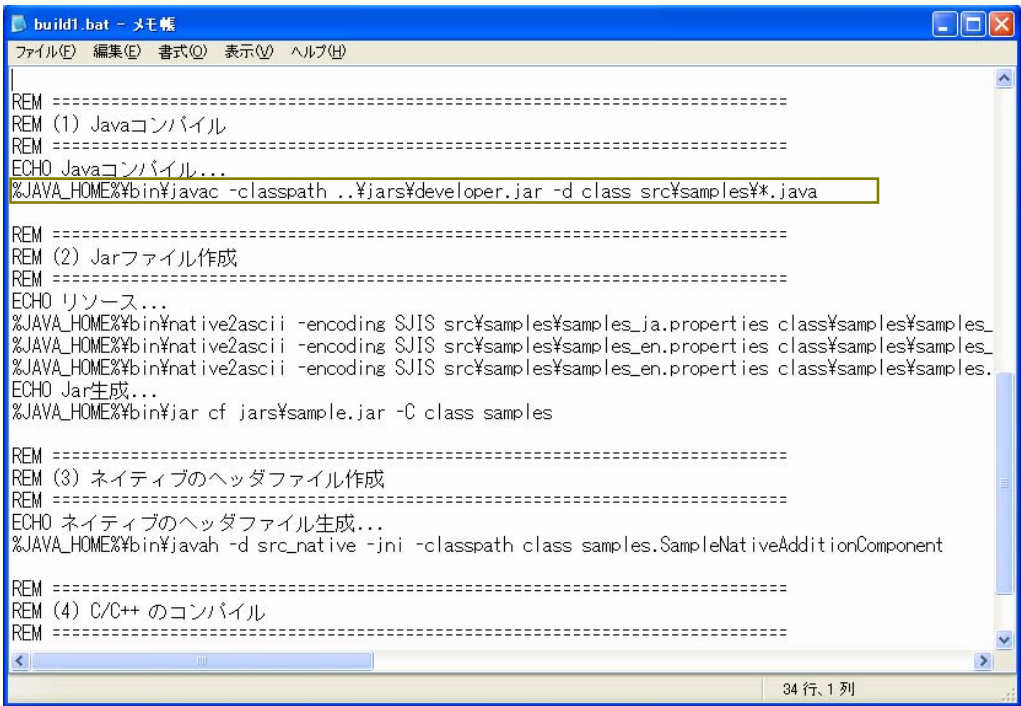
REM =====
REM 環境設定
REM =====
ECHO 環境設定...
set JAVA_HOME=c:¥j2sdk1.4.2_10
REM set MINGW_HOME=c:¥mingw

26 行, 5 列
```

<sup>1</sup> 例えば、[http://java.sun.com/products/archive/j2se/1.4.2\\_10/index.html](http://java.sun.com/products/archive/j2se/1.4.2_10/index.html) からダウンロード。

画面下へ半分ほど移動したところに以下の記述があります。

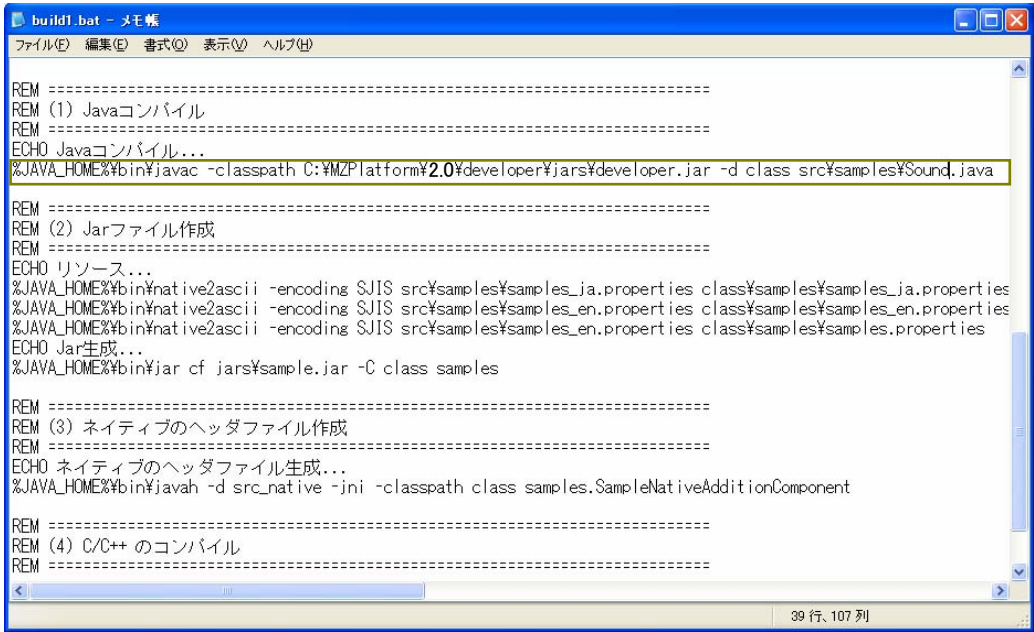
```
%JAVA_HOME%\bin\javac -classpath ..\jars\developer.jar -d class src\samples\*.java
```



「..」と「\*」の部分を以下の青色文字に変更します。

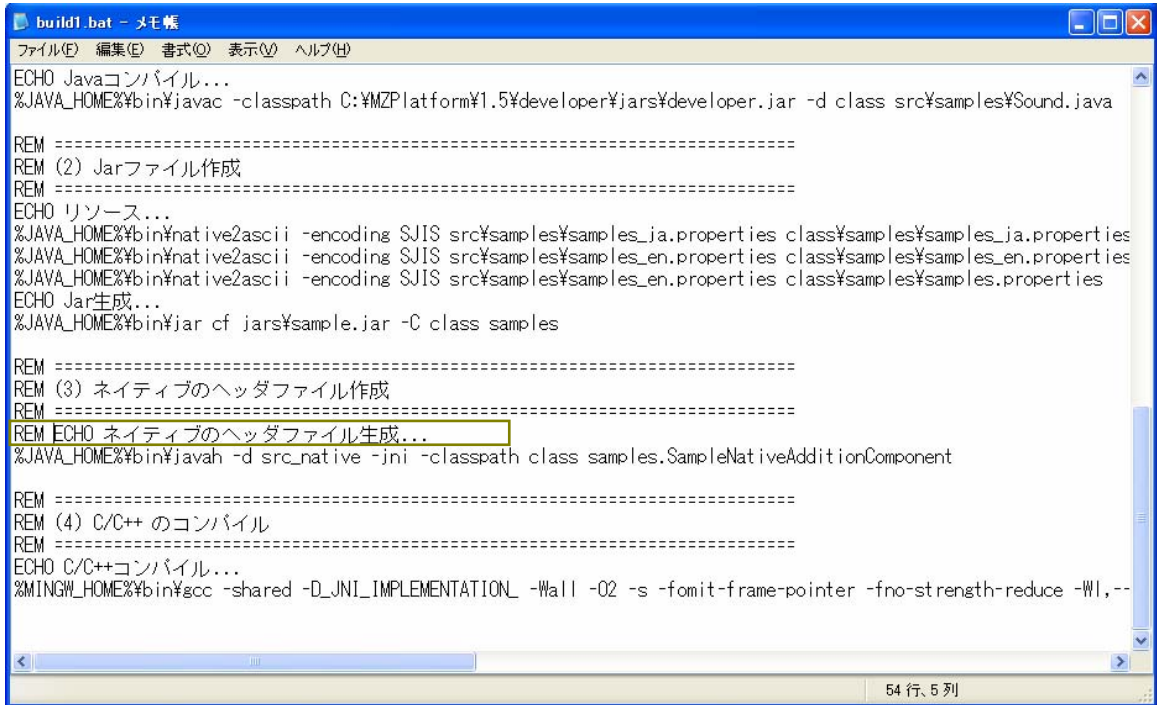
```
%JAVA_HOME%\bin\javac -classpath C:\MZPlatform\2.0\developer\jars\developer.jar -d class src\samples\Sound.java
```

(注) 入力は半角で行います。



画面を少し下へ移動すると「ECHO ネイティブのヘッダファイル生成...」の記述がありますが、この命令文は実行しないのでコメント（注釈）文にします。以下の青色部分を追加します。

**REM△ECHO ネイティブのヘッダファイル生成...**  
**(注) △部分は半角スペースを入力します。**



```
build1.bat - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
ECHO Javaコンパイル...
%JAVA_HOME%\bin\javac -classpath C:\MZPlatform¥1.5¥developer¥jars¥developer.jar -d class src¥samples¥Sound.java

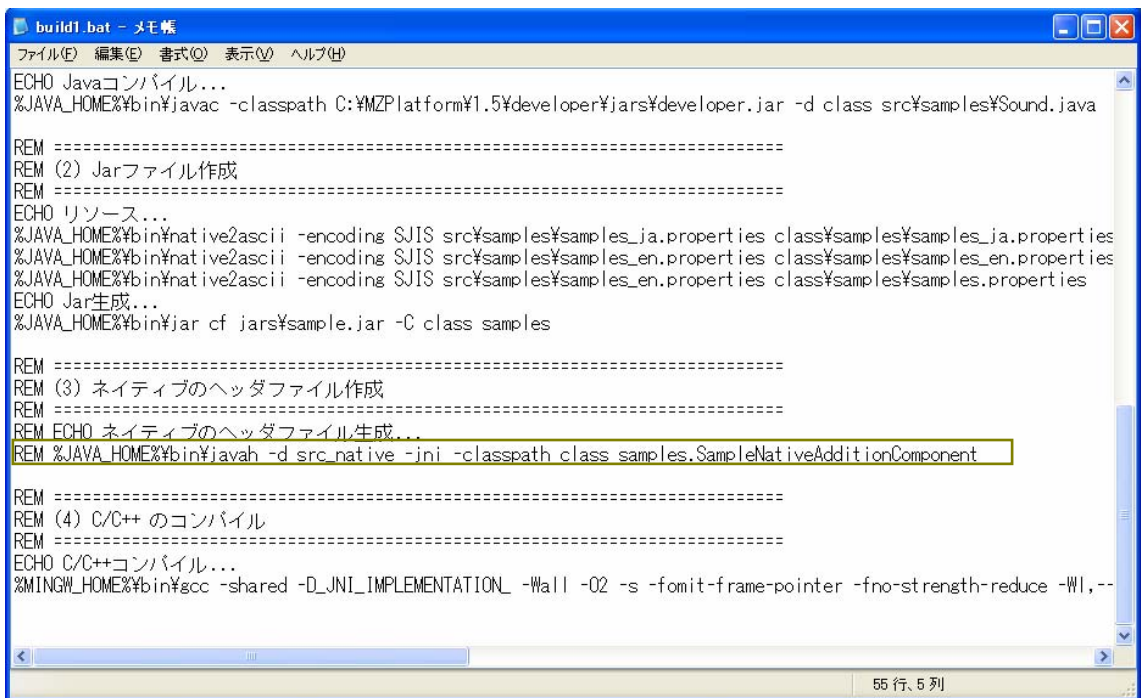
REM =====
REM (2) Jarファイル作成
REM =====
ECHO リソース...
%JAVA_HOME%\bin\native2ascii -encoding SJIS src¥samples¥samples_ja.properties class¥samples¥samples_ja.properties
%JAVA_HOME%\bin\native2ascii -encoding SJIS src¥samples¥samples_en.properties class¥samples¥samples_en.properties
%JAVA_HOME%\bin\native2ascii -encoding SJIS src¥samples¥samples_en.properties class¥samples¥samples.properties
ECHO Jar生成...
%JAVA_HOME%\bin\jar cf jars¥sample.jar -C class samples

REM =====
REM (3) ネイティブのヘッダファイル作成
REM =====
REM ECHO ネイティブのヘッダファイル生成...
%JAVA_HOME%\bin\javah -d src_native -jni -classpath class samples.SampleNativeAdditionComponent

REM =====
REM (4) C/C++ のコンパイル
REM =====
ECHO C/C++コンパイル...
%MINGW_HOME%\bin\gcc -shared -D_JNI_IMPLEMENTATION_ -Wall -O2 -s -fomit-frame-pointer -fno-strength-reduce -Wl,--
```

次に「%JAVA\_HOME%\bin\javah... (以下省略)」の記述がありますが、この命令文は実行しないのでコメント（注釈）文にします。以下の青色部分を追加します。

**REM△%JAVA\_HOME%\bin\... (以下省略)**  
**(注) △部分は半角スペースを入力します。**



```
build1.bat - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
ECHO Javaコンパイル...
%JAVA_HOME%\bin\javac -classpath C:\MZPlatform¥1.5¥developer¥jars¥developer.jar -d class src¥samples¥Sound.java

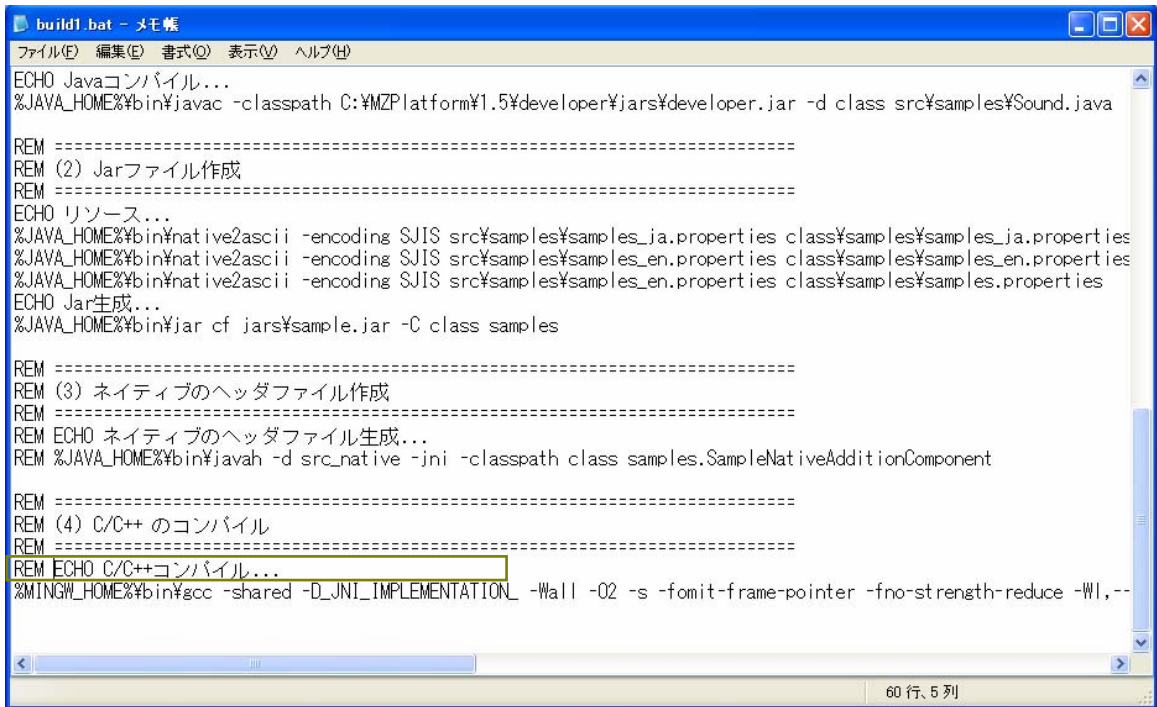
REM =====
REM (2) Jarファイル作成
REM =====
ECHO リソース...
%JAVA_HOME%\bin\native2ascii -encoding SJIS src¥samples¥samples_ja.properties class¥samples¥samples_ja.properties
%JAVA_HOME%\bin\native2ascii -encoding SJIS src¥samples¥samples_en.properties class¥samples¥samples_en.properties
%JAVA_HOME%\bin\native2ascii -encoding SJIS src¥samples¥samples_en.properties class¥samples¥samples.properties
ECHO Jar生成...
%JAVA_HOME%\bin\jar cf jars¥sample.jar -C class samples

REM =====
REM (3) ネイティブのヘッダファイル作成
REM =====
REM ECHO ネイティブのヘッダファイル生成...
REM %JAVA_HOME%\bin\javah -d src_native -jni -classpath class samples.SampleNativeAdditionComponent

REM =====
REM (4) C/C++ のコンパイル
REM =====
ECHO C/C++コンパイル...
%MINGW_HOME%\bin\gcc -shared -D_JNI_IMPLEMENTATION_ -Wall -O2 -s -fomit-frame-pointer -fno-strength-reduce -Wl,--
```

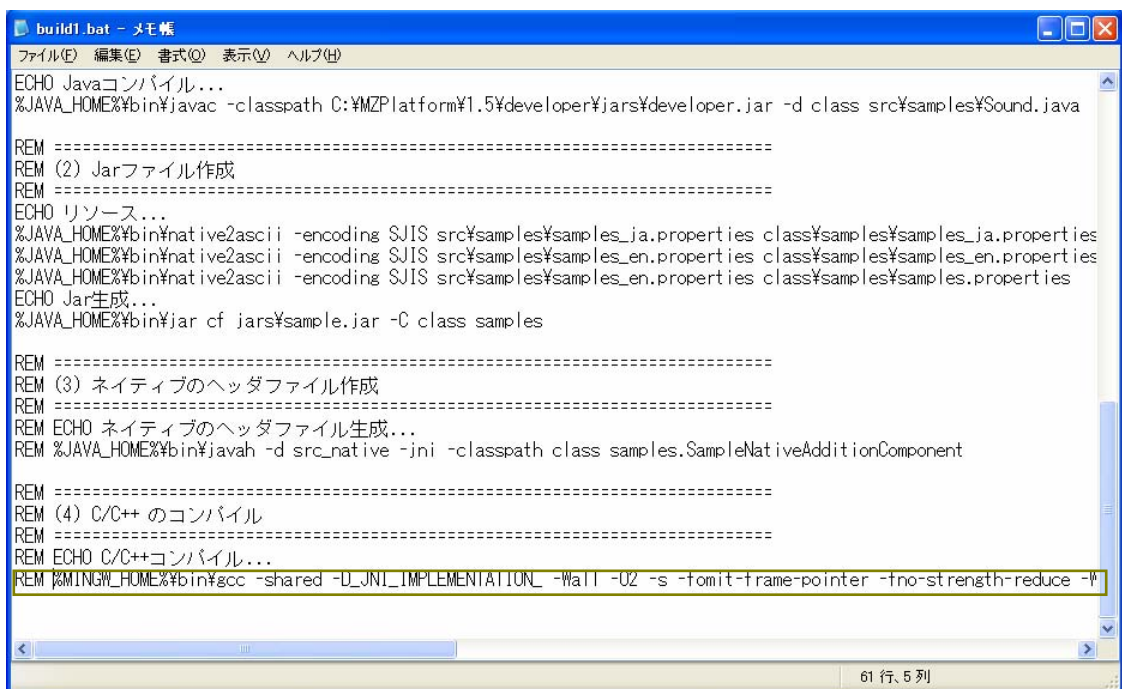
最下行から2行目に「ECHO C/C++コンパイル...」の記述がありますが、この命令文は実行しないのでコメント（注釈）文にします。以下の青色部分を追加します。

```
REM△ECHO C/C++コンパイル...
(注) △部分は半角スペースを入力します。
```

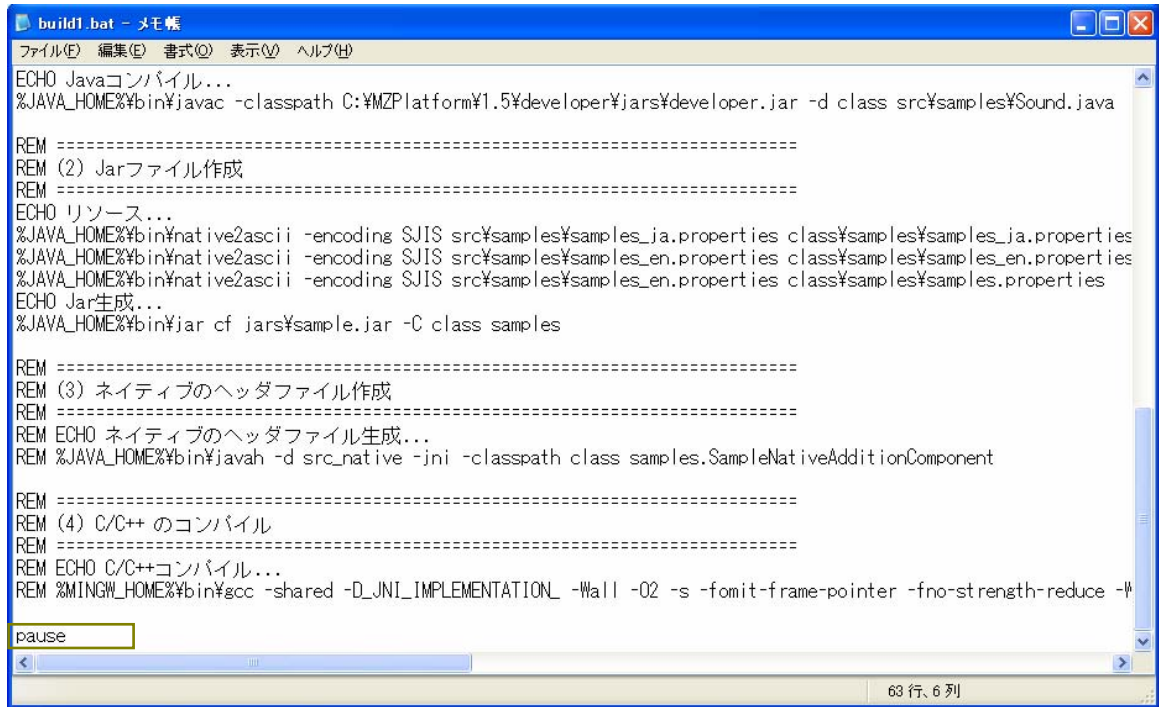


最下行に「%MINGW\_HOME%¥bin¥gcc...（以下省略）」の記述がありますが、この命令文は実行しないのでコメント（注釈）文にします。以下の青色部分を追加します。

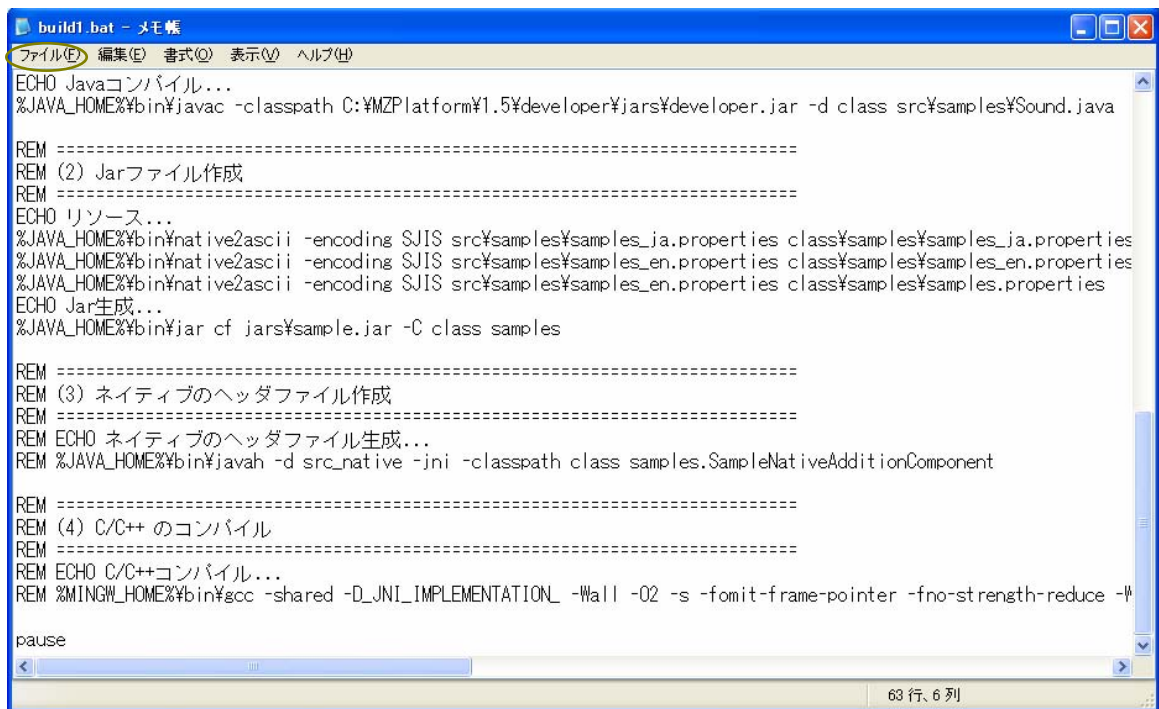
```
REM△%MINGW_HOME%¥bin¥gcc...（以下省略）
(注) △部分は半角スペースを入力します。
```



最下行の記述の次行に改行を挿入します。改行を挿入した次行に「pause」の記述を行います。

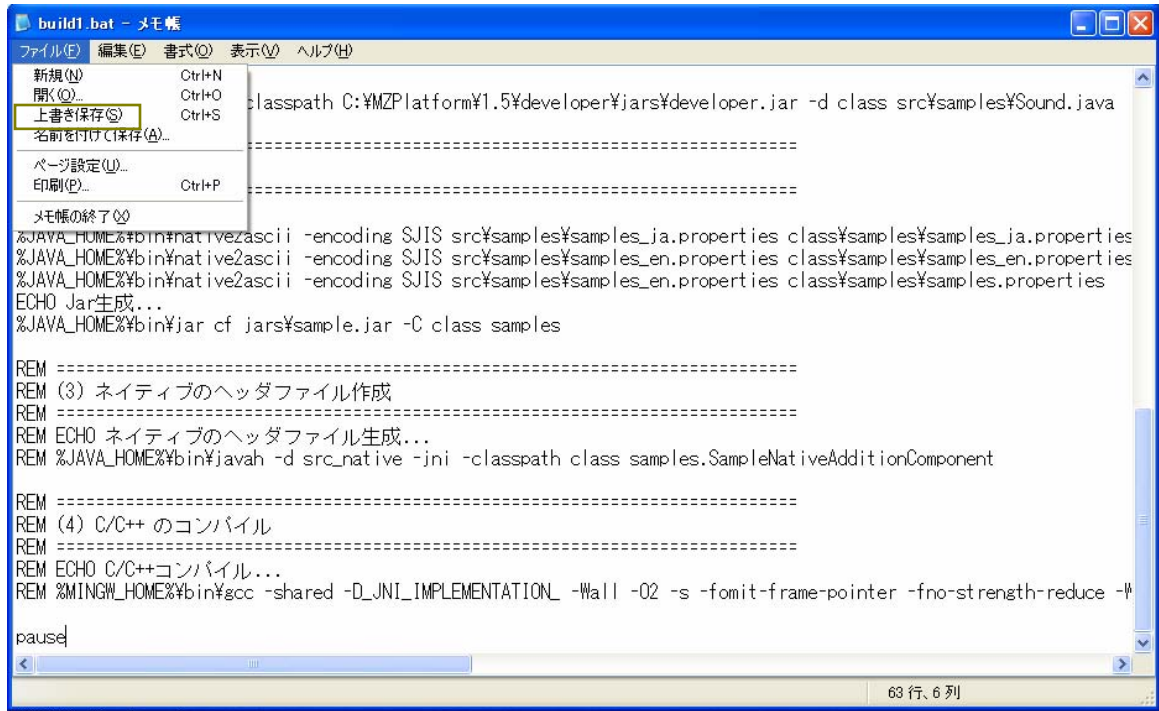


ここで行った設定を保存します。メニューバーの「ファイル(F)」を左クリックします。

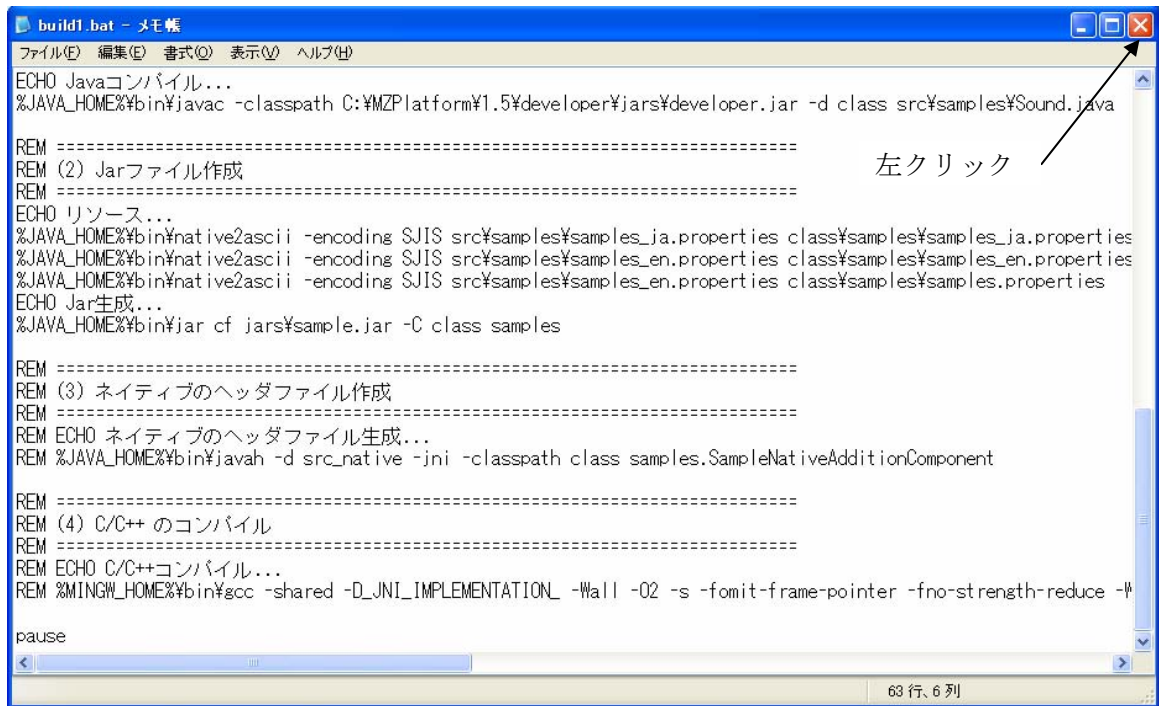




一覧から「上書き保存(S)」を左クリックします。

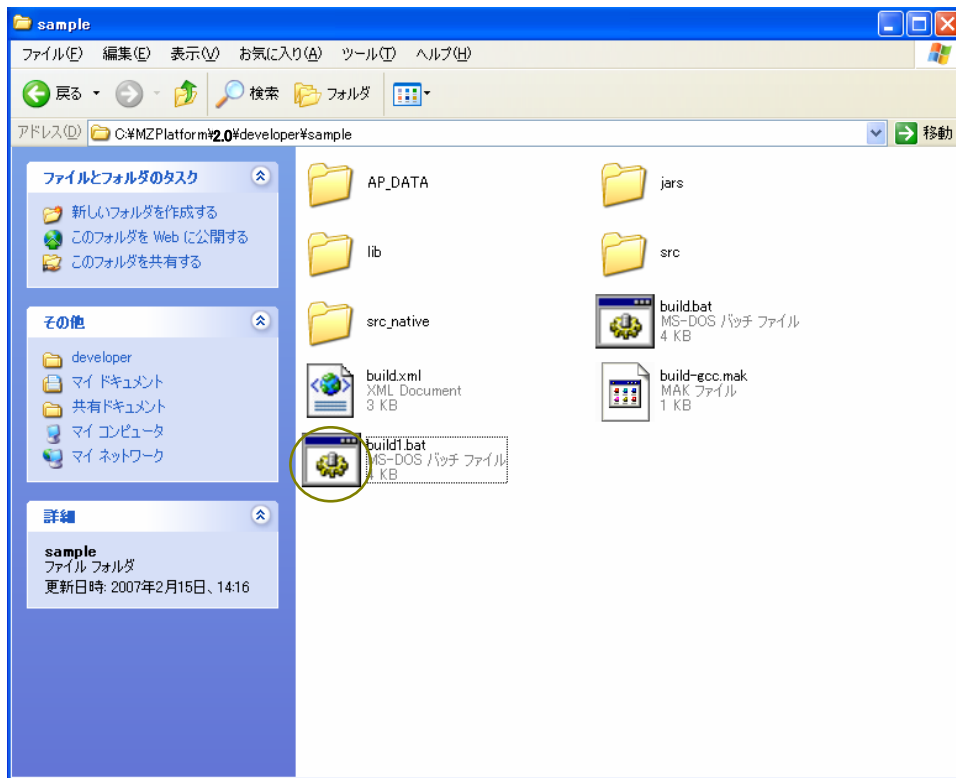


「×」ボタンを左クリックし、「build1.bat」ファイルを閉じます。

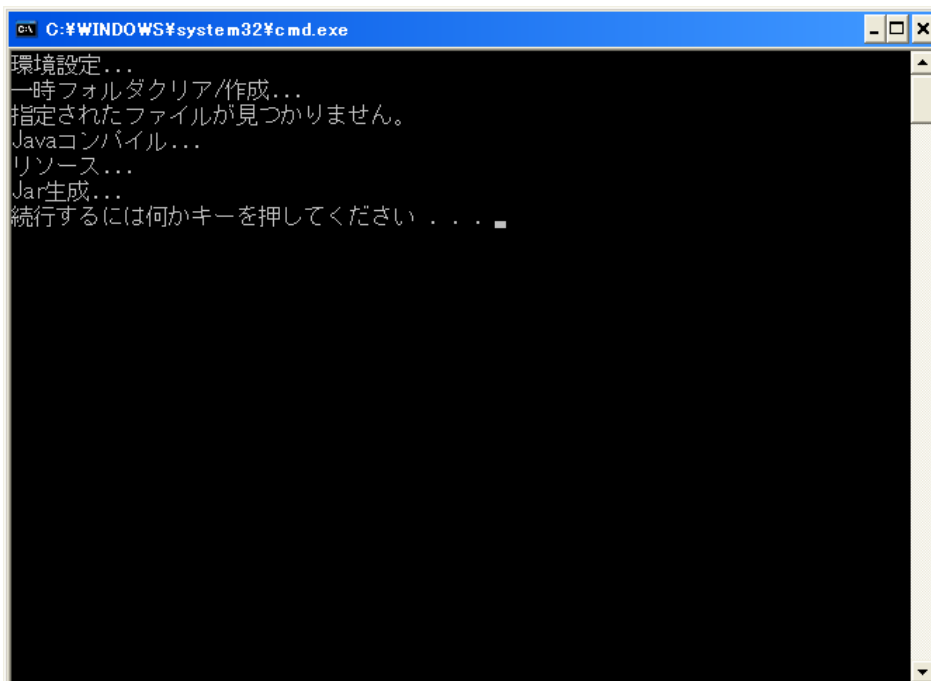


## 手順4 バッチファイルの実行

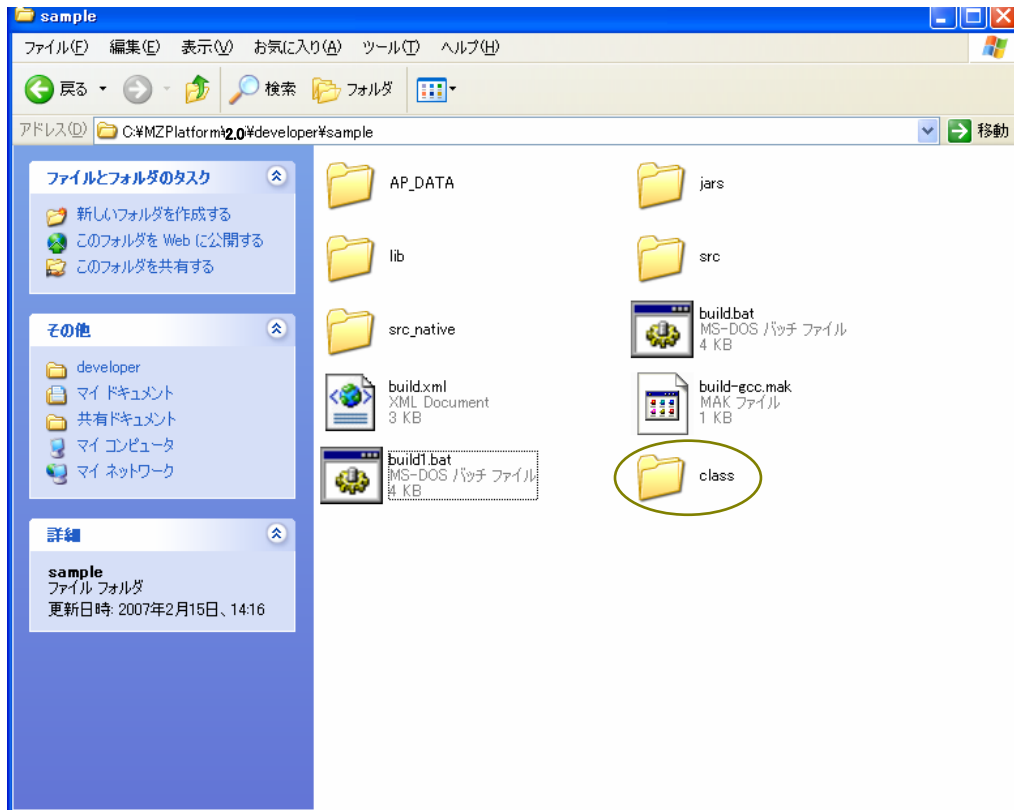
「build1.bat」ファイルのアイコンをダブルクリックします。



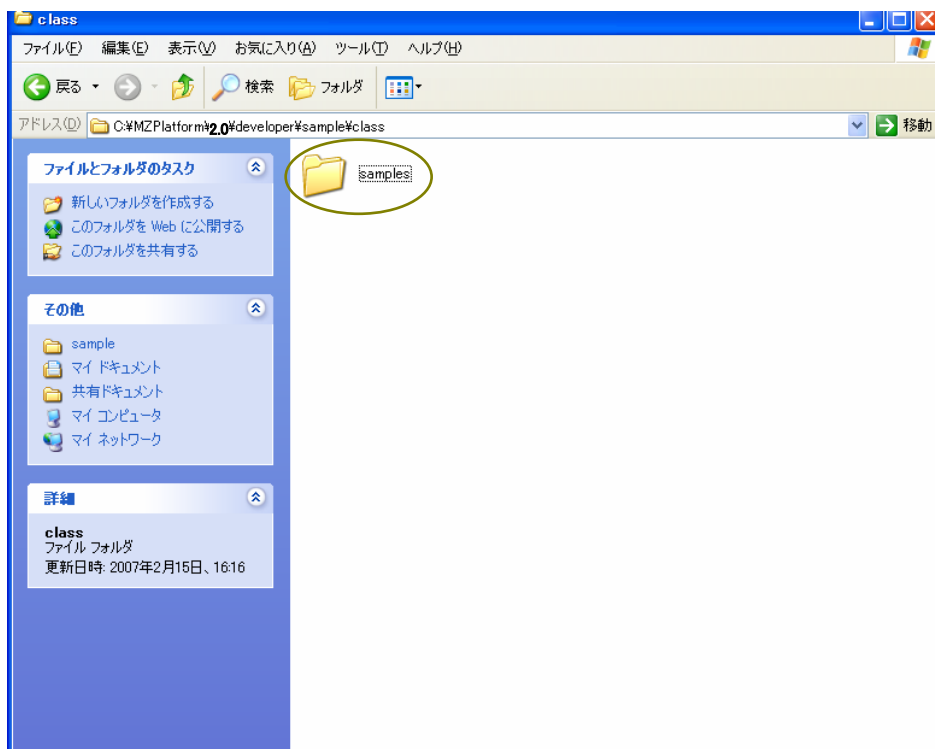
以下の画面が表示されます。[Enter]キーを押してください。(もしエラーメッセージが表示されたらどこか設定を間違っています。もう一度手順を確認してください。)



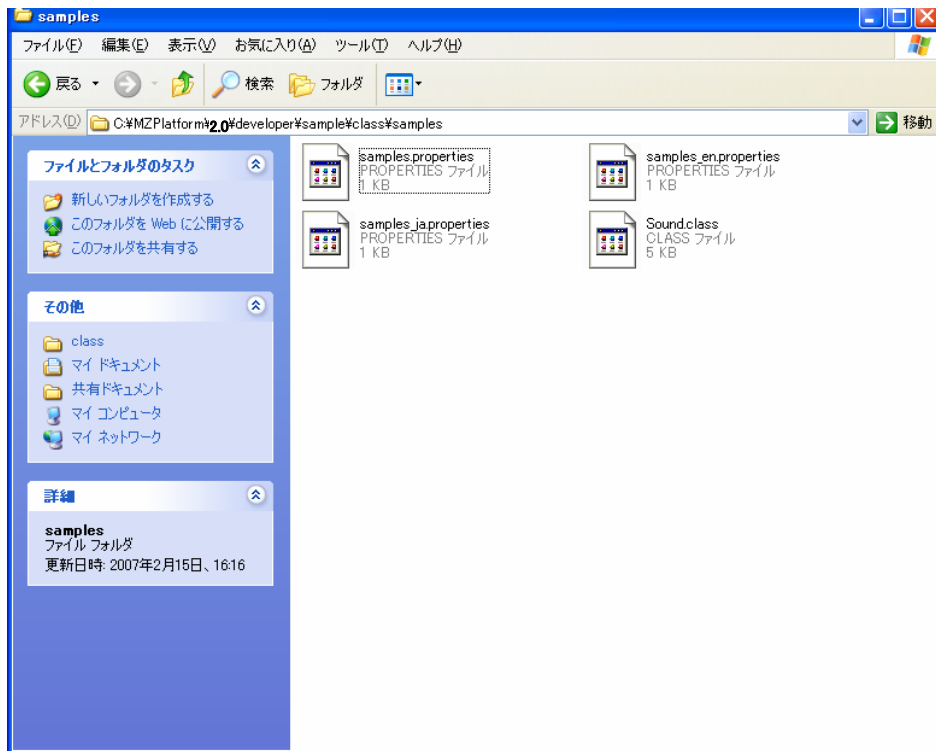
「class」フォルダが作成されたことが確認できます。



「スタート」→「マイコンピュータ」→「ローカルディスク(C:)」→「MZPlatform」→「2.0」→「developer」→「sample」→「class」とたどります。「sample」フォルダが作成されたことが確認できます。

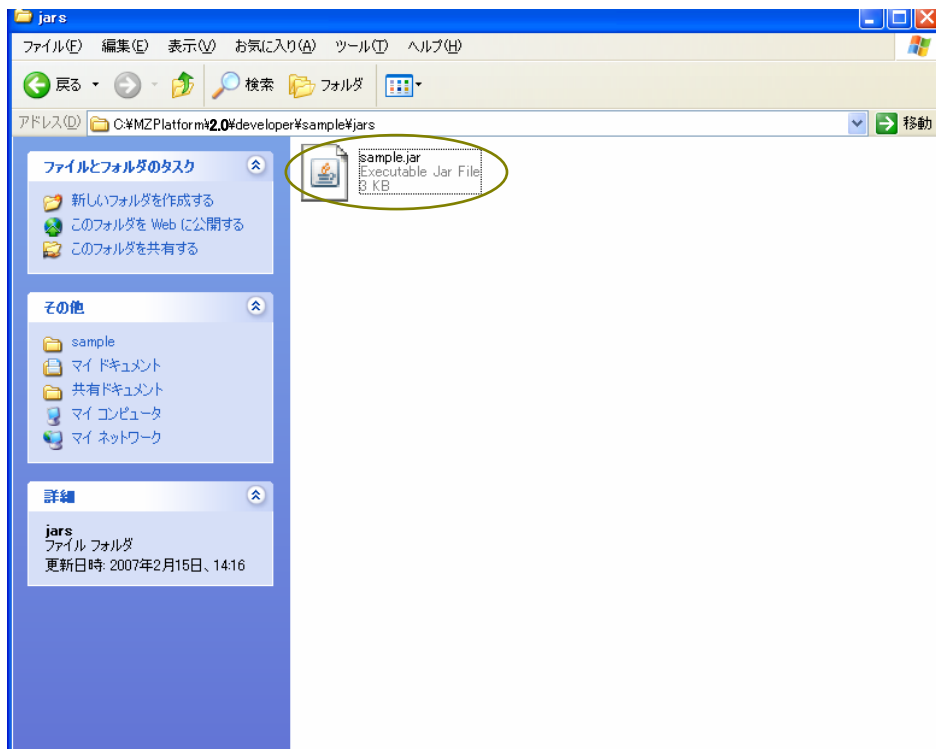


「samples」フォルダのアイコンをダブルクリックします。フォルダ内に「samples.properties」、「samples\_en.properties」、「samples\_ja.properties」、「Sound.class」ファイルが作成されたことが確認できます。

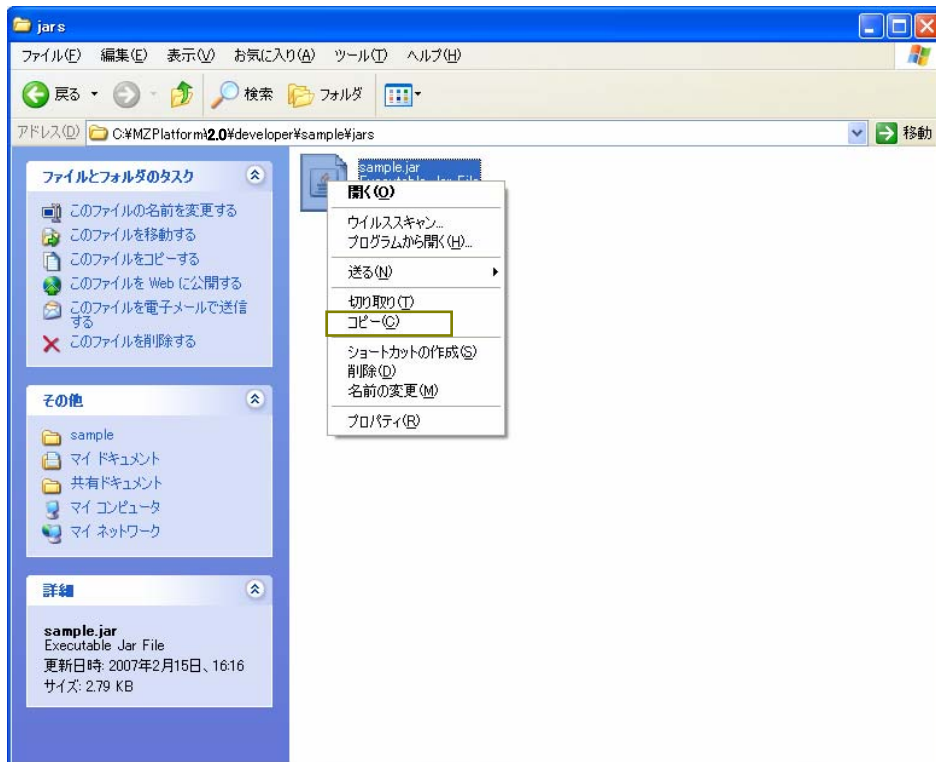


## 手順5 「sample.jar」ファイルの確認

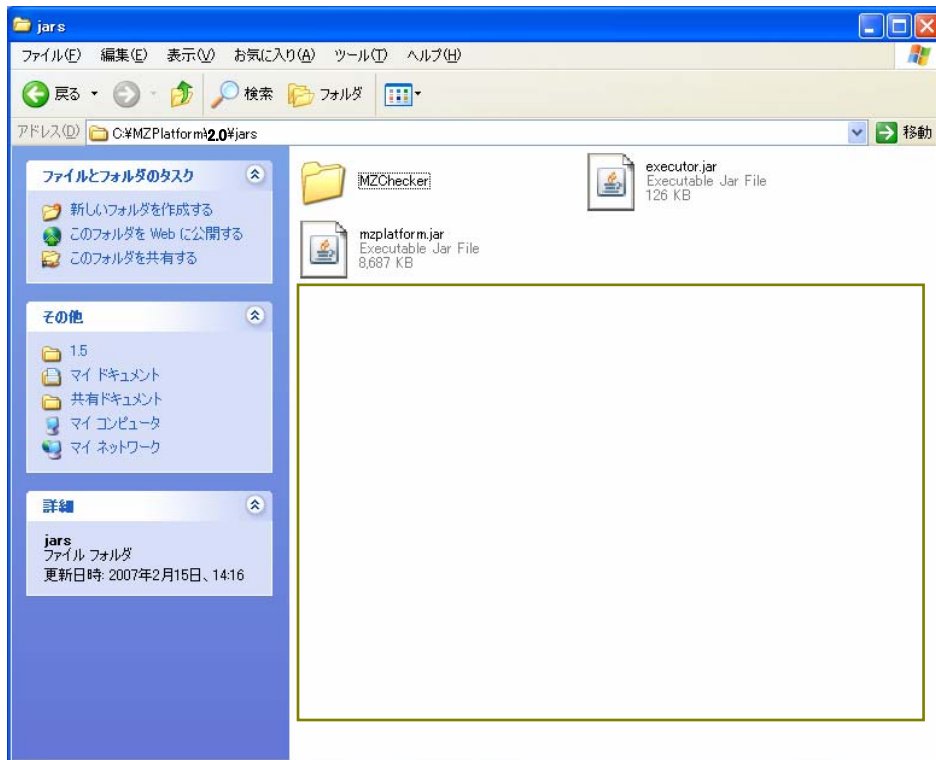
「jars」フォルダ内に「sample.jar」ファイルが作成されているかを確認します。「スタート」→「マイコンピュータ」→「ローカルディスク(C:)」→「MZPlatform」→「2.0」→「developer」→「sample」→「jars」とたどります。「sample.jar」ファイルが作成されているのが確認できます。



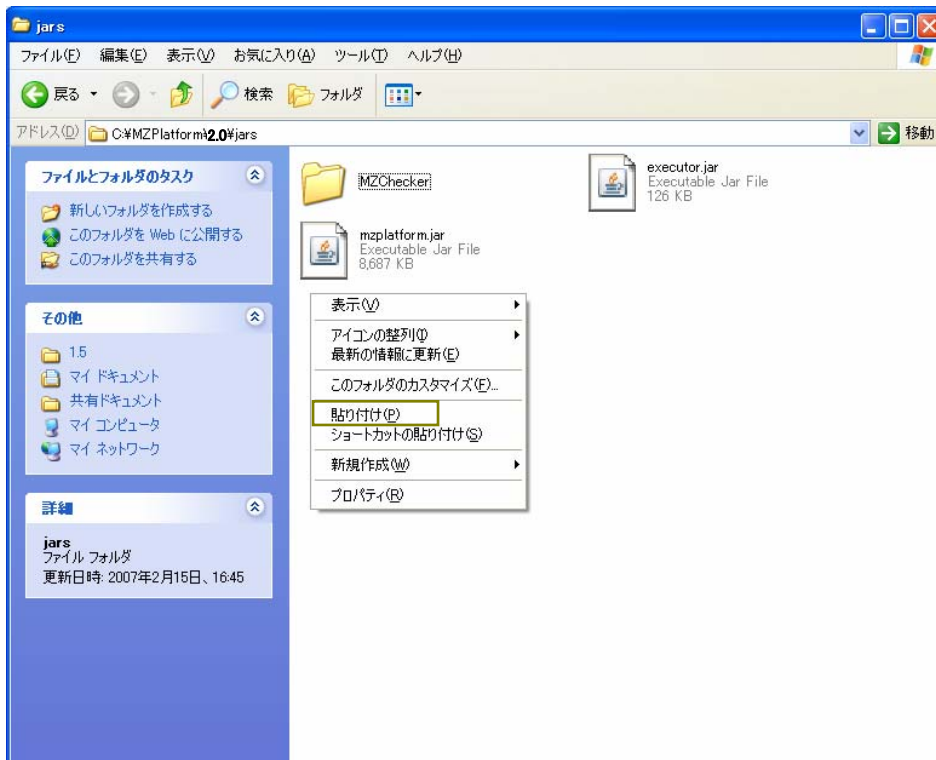
「sample.jar」ファイルを移動させます。「sample.jar」ファイルのアイコン上で右クリックし、[コピー(C)]を左クリックします。



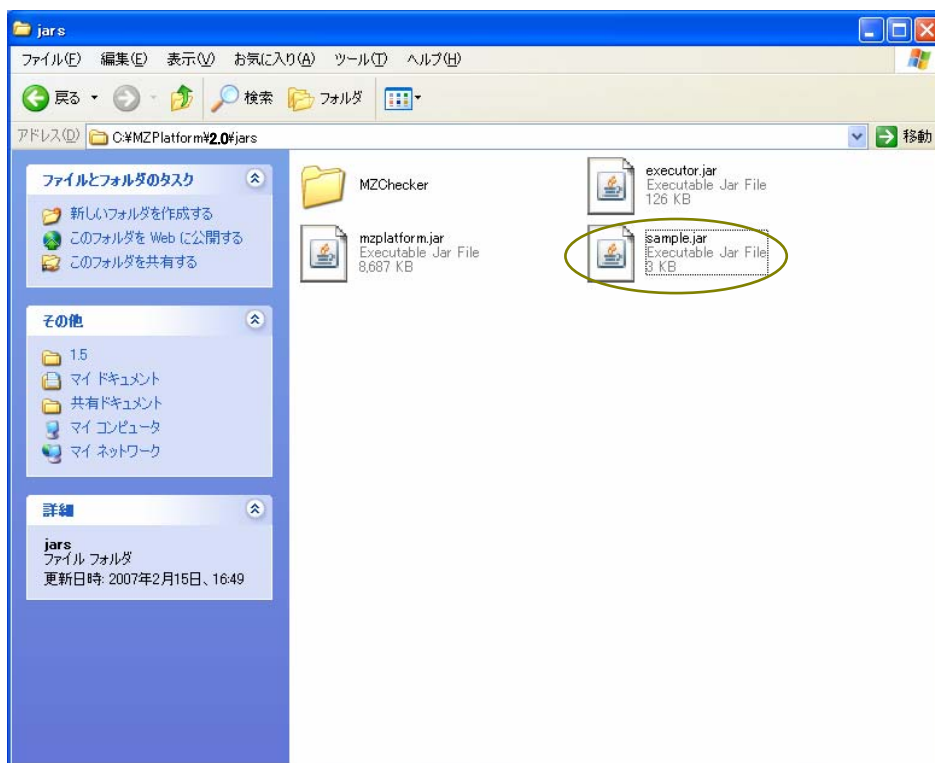
「スタート」→「マイコンピュータ」→「ローカルディスク(C:)」→「MZPlatform」→「2.0」→「jars」とたどります。下図の囲み内で右クリックします。



[貼り付け(P)]を左クリックします。

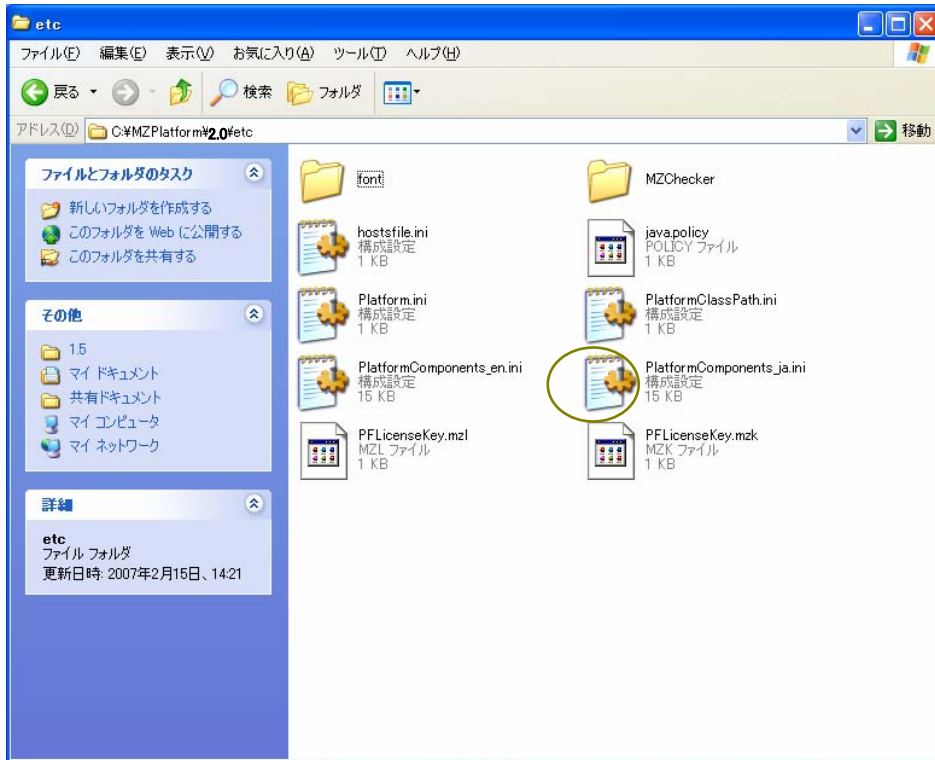


「sample.jar」ファイルが追加されたことが確認できます。



## 手順6 コンポーネント登録手続き (その1)

「スタート」→「マイコンピュータ」→「ローカルディスク(C:)」→「MZPlatform」→「2.0」→「etc」とたどります。「PlatformComponents\_ja.ini」ファイルのアイコンをダブルクリックします。

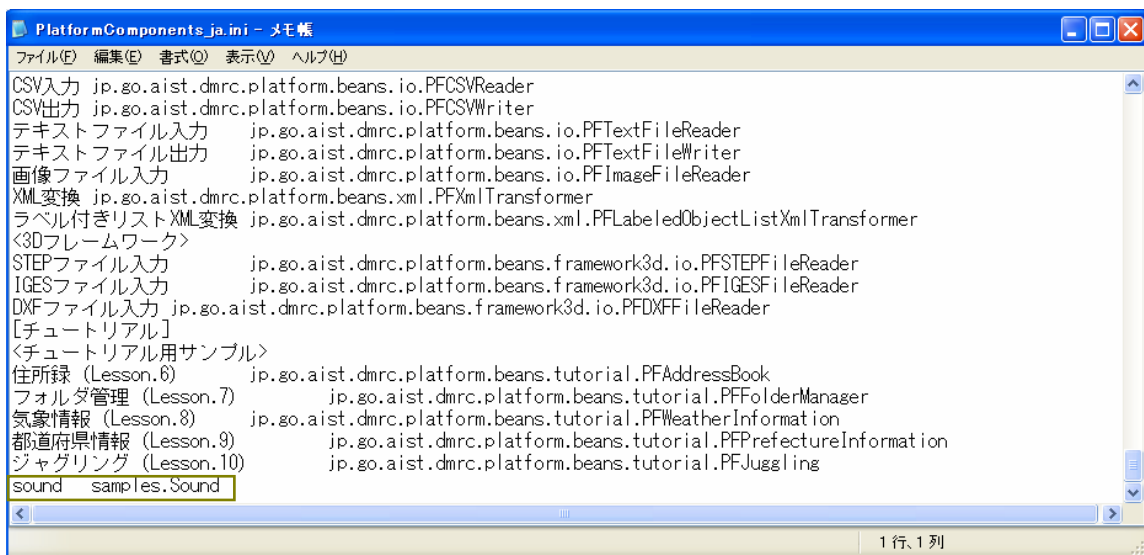


最下行に「ジャグリング(Lesson.10) jp.go.aist.dmrc... (以下省略)」の記述があります。次行に以下の記述を行います。

sound△samples.Sound

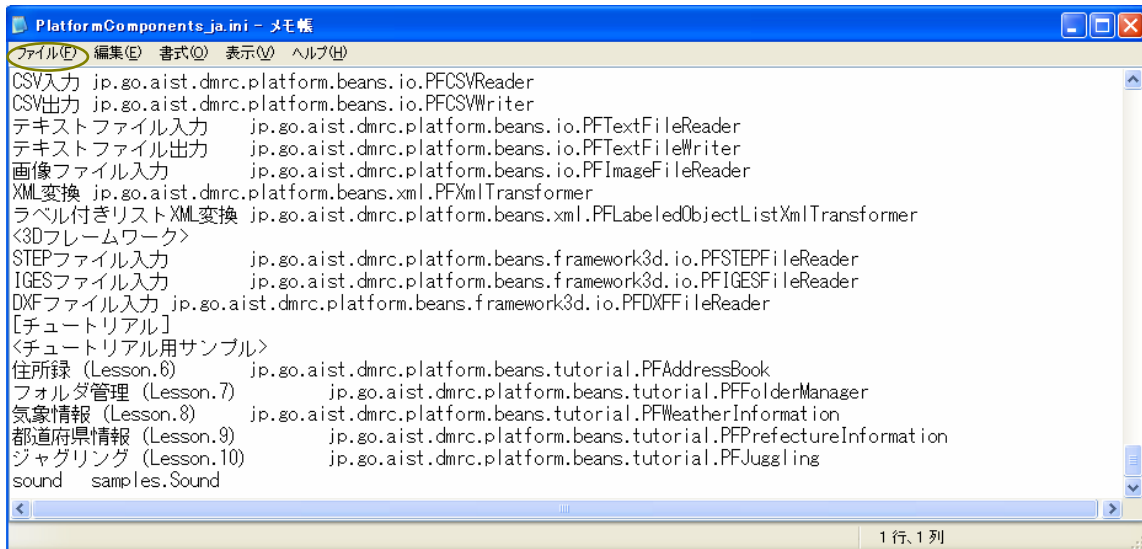
(注1) 入力は半角で行います。

(注2) △部分は[Tab]キーを1回押して、スペースを挿入します。

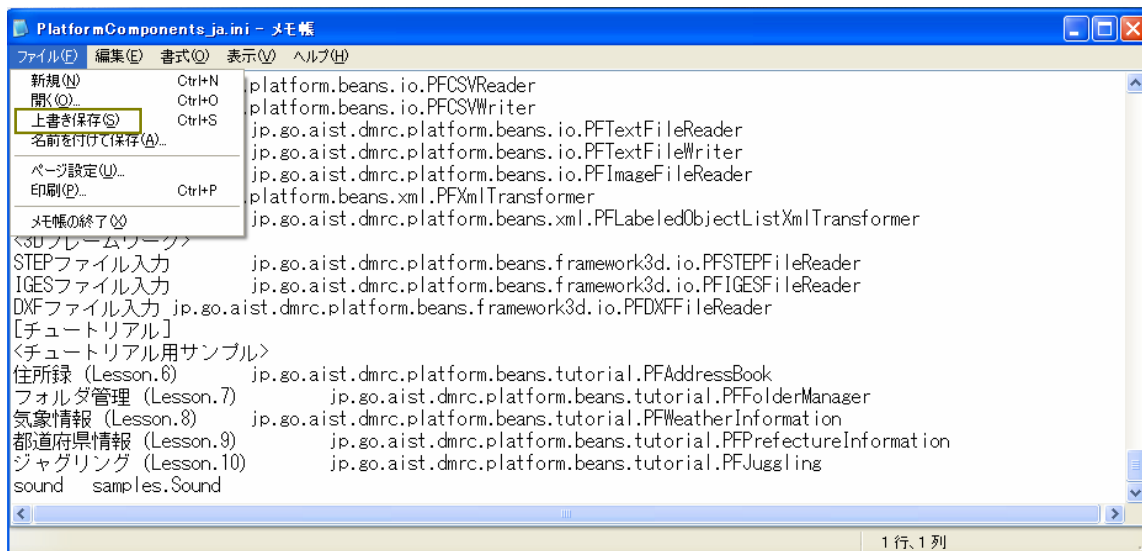




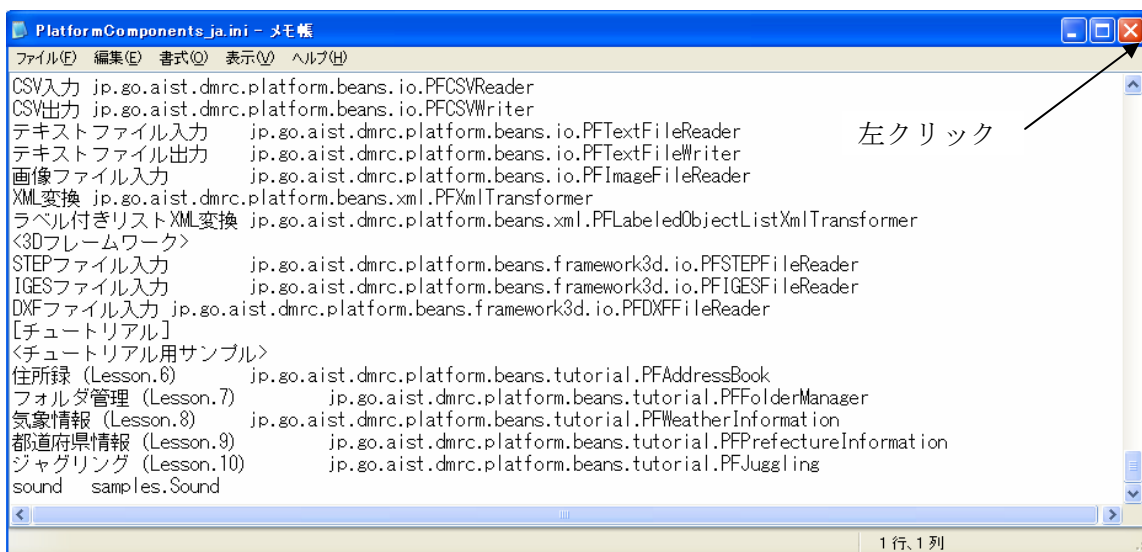
ここで行った設定を保存します。メニューバーの[ファイル(F)]を左クリックします。



[上書き保存(S)]を左クリックします。

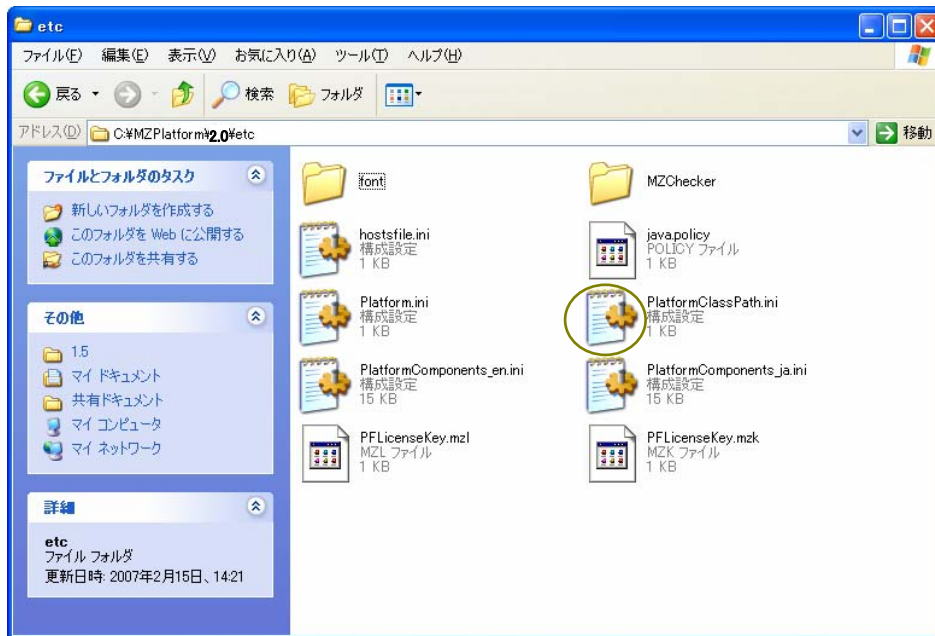


「×」ボタンを左クリックし、「PlatformComponents\_ja.ini」ファイルを閉じます。



## 手順7 コンポーネント登録手続き (その2)

「スタート」→「マイコンピュータ」→「ローカルディスク(C:)」→「MZPlatform」→「2.0」→「etc」とたどります。「PlatformClassPath.ini」ファイルのアイコンをダブルクリックします。

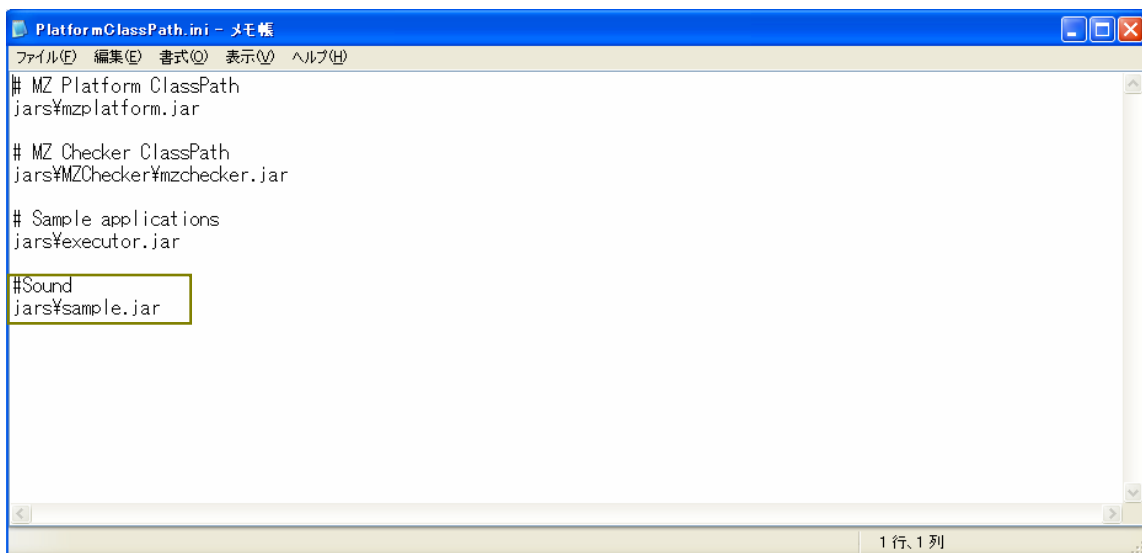


最下行に「jars¥executor.jar」の記述があります。次行に空白行を挿入し、以下の記述を追加します。

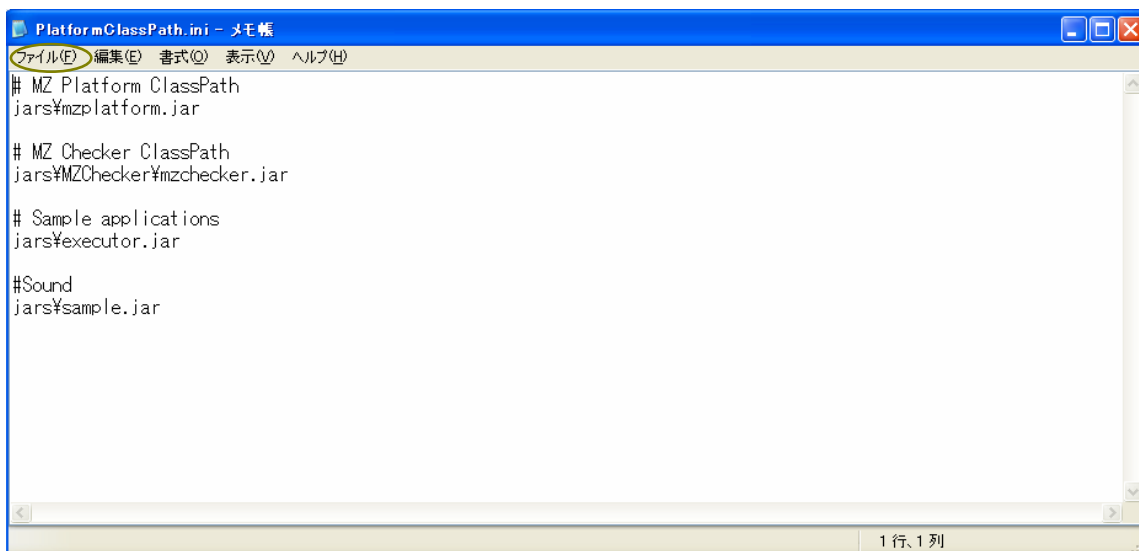
```
#Sound
```

```
jars¥sample.jar
```

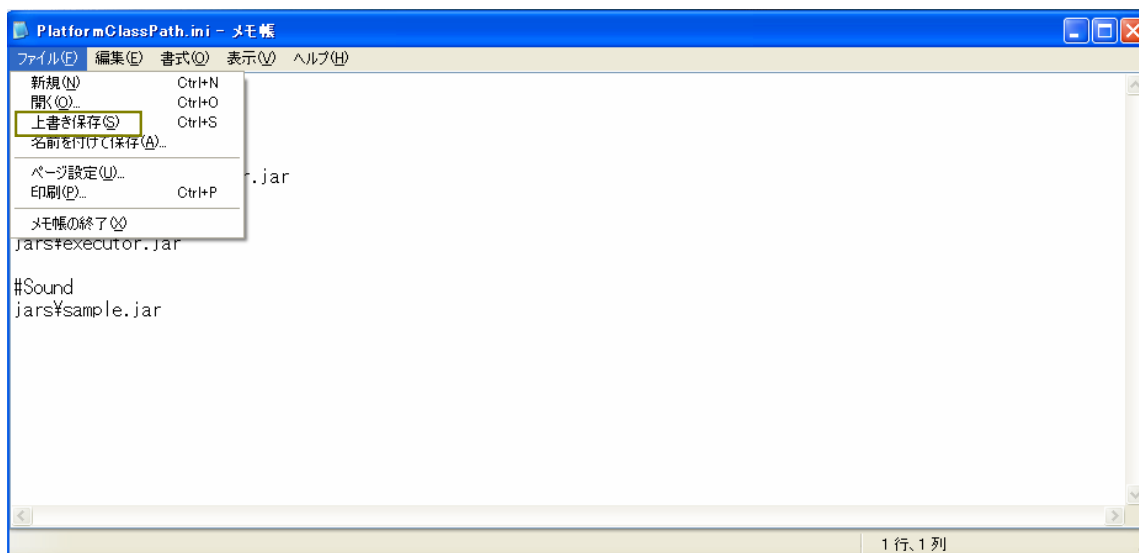
(注) 入力は半角で行います。



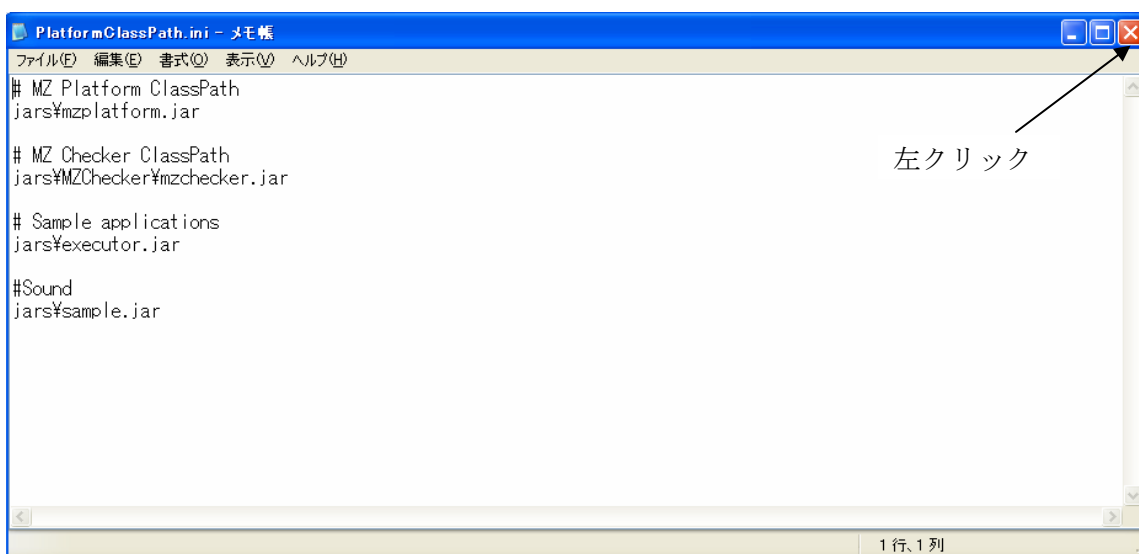
ここで行った設定を保存します。メニューバーの[ファイル(F)]を左クリックします。



[上書き保存(S)]を左クリックします。

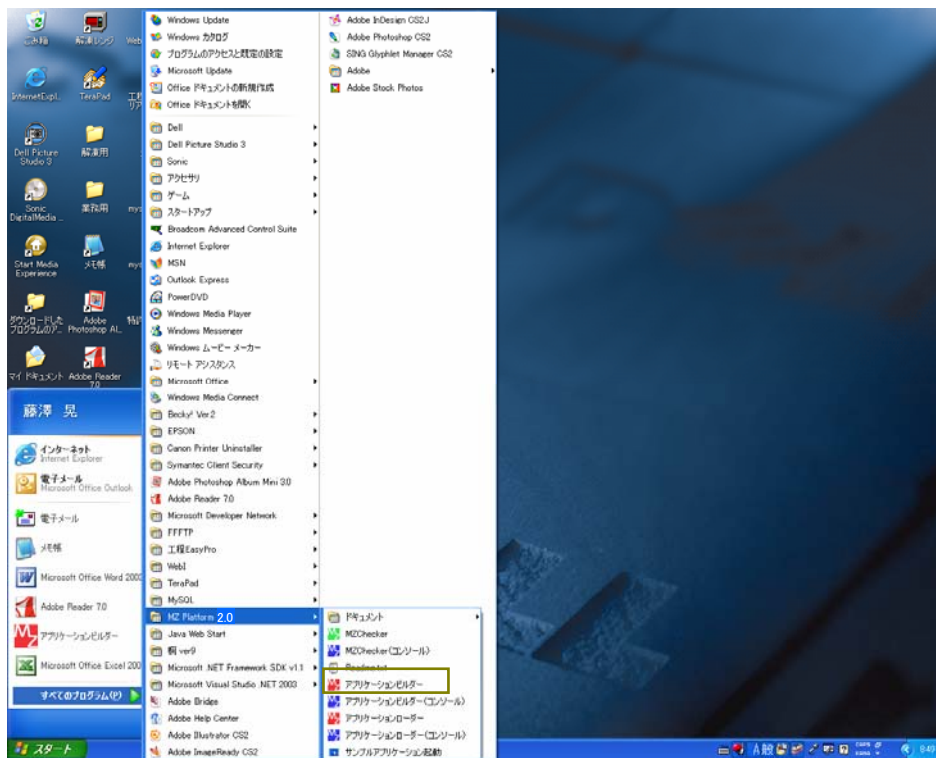


「×」ボタンを左クリックし、「PlatformClassPath.ini」ファイルを閉じます。

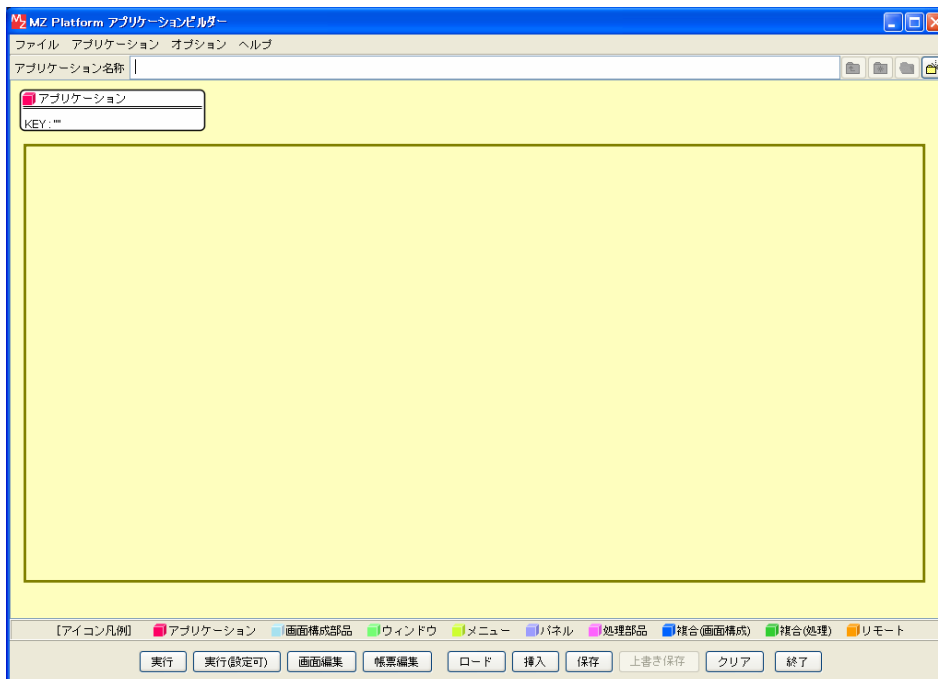


### 手順8 コンポーネント一覧に登録されているかの確認

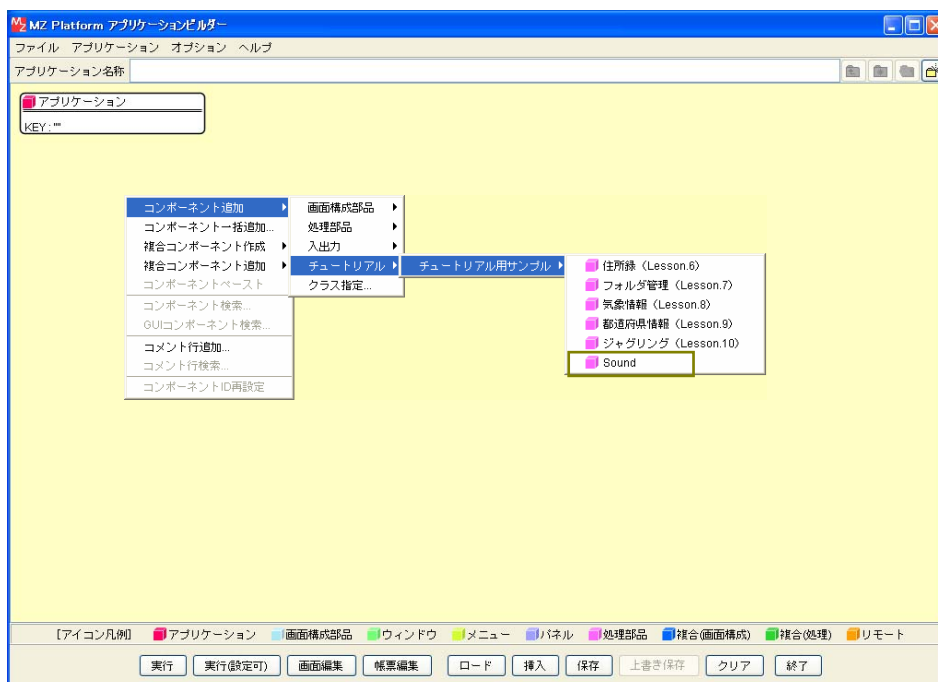
「スタート」→「すべてのプログラム(P)」→「MZ Platform 2.0」とどり、「アプリケーションビルダー」を左クリックします。



「アプリケーションビルダー」画面が表示されます。下図の囲み内で右クリックします。

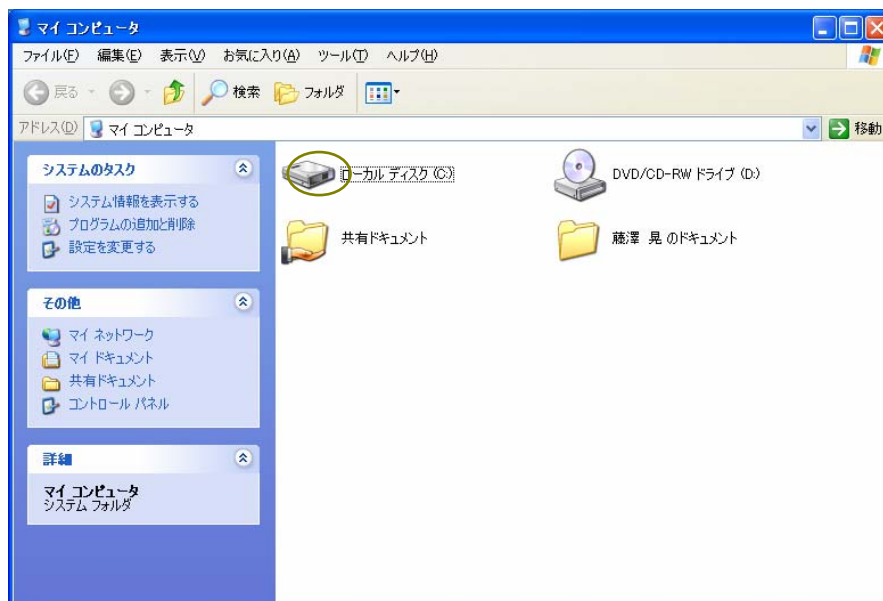


「コンポーネント追加」 → 「チュートリアル」 → 「チュートリアル用サンプル」とたどります。末尾に「sound」コンポーネントが登録されていることが確認できます。



## Java 開発環境 (JDK) インストールフォルダの確認方法

「スタート」→「マイコンピュータ」とたどります。「ローカルディスク(C:)」アイコンをダブルクリックします。



「ローカルディスク(C:)」画面が表示されます。「j2sdk1.4.2\_10」フォルダが確認できます。

