3-12.プロパティファイル

プロパティファイル

はじめに

プロパティとは、クラスの属性値・設定値のことで、
プログラムのフィールド変数やメソッドとは別のものになります。

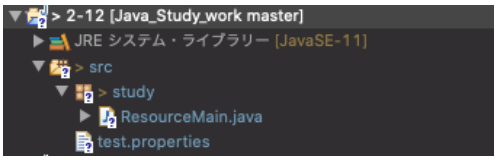
javaには、プロパティファイル（拡張子は `.properties` ）との
入出力ができるクラスが用意されています。
なので、javaファイルからプロパティファイルを読み込んだり、
プロパティファイルへ書き込んだりする事が可能です。

この章では、javaにおけるプロパティファイルの基本的な使い方を学んで行きましょう！

Step1: プロパティファイルの記述

> **例題**
> 2-12というプロジェクトを作成し、下記画像のようにファイルを作成して下さい。

`test.properties`は、`src`を右クリック→新規→その他→一般→ファイル　で作成できます。




```
// プロパティファイルの記述
driver = org.postgresql.Driver
connection : jdbc:postgresql://localhost:5434/lesson_db?useUnicode=true&characterEncoding=utf8
userId : postgres
pass = postgres
```

> postgreSQLインストール時にポート番号を「5432」に設定している人は、connectionキーの値の一部分 **localhost:5434/〜** では正しく動作しません。 **localhost:5432/〜** と記述してください。

プロパティファイルの書き方は`キー = 値` or `キー : 値` という形になります。

> **補足**

例題では`=`・`:`どちらも使っていますが、どちらかで統一して記述しましょう！

Step2: ResourceBundle

ResourceBundleクラスを使用すると、簡単にプロパティファイルから値を読み込むことができます。

> **例題**【ResourceMain.java】

?

```
package study;

import java.util.ResourceBundle;

public class ResourceMain {

    public static void main(String args[]) {
        ResourceBundle bundle = ResourceBundle.getBundle("任意のファイル名（拡張子なし）");
        String value = bundle.getString("任意のキー");
        System.out.println(value);
    }
}
```

> **getBundle()**

`getBundle()`メソッドでプロパティファイルを読み込みます。

このときプロパティファイル名だけ指定し、ファイルの拡張子はいりません。
今回の例ですと、`getBundle("test")` とすれば読み込みます。

> **getString()**

`getString()`メソッドで、プロパティファイルに記述されている値が取得することができます。
取得の仕方は、`getString()`の引数には、プロパティファイルのキーを渡してあげます。
そうすると、そのキーに紐付いた値が取得できます。
今回の場合ですと、`getString(pass)` とすれば、`postgres`が取得できます。

このようにして、プロパティファイルの値をjavaの処理で扱うことができます。
基本的な構造は、以上になります！

> **補足**

プロパティファイルには、 **コード上に持たせないような情報** を記述します。
もっと言うと、プロパティファイルに記述しておけば運用・メンテナンスが
簡単になるものをプロパティファイルに記述します。

複数のjavaソースで使いたい定数などが考えられます。
それはデータベースのIDやPassWordなどですね。

仮に、IDが変わったとしても、
プロパティファイルの情報を変更するだけで参照している、すべてのjavaソースに反映されます。

プロパティファイルは、そのようにして活用されています。

Step3: 文字コード

先ほど作ったプロパティファイルに、 **日本語** を追加してみよう。
日本語であればなんでもよいです。

> **例題**

```
//プロパティファイル
name = "鈴木"
```

> **例題**

下記の文で、プロパティファイルを呼び出し、日本語を出力してみよう。

```
package study;

import java.util.ResourceBundle;

public class ResourceMain {

    public static void main(String args[]) {
        ResourceBundle bundle = ResourceBundle.getBundle("test");
        String value = bundle.getString("name");
        System.out.println(value);
    }
}
```

java8以前のバージョンであれば、下記のように暗号が出力されます。

```
"ä¸æ¬¡"
```

これが俗に言う、 **文字化け** です。

例：文字化けを改善するコード

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.Properties;

/**
 * UTF-8 エンコーディングされたプロパティファイルを {@link Properties} クラスで取り扱う。
 */
public class ResourceMainEncoding {

    static Properties loadUtf8Properties(String resourceName) throws IOException {

        try (InputStream is = ResourceMainEncoding.class.getResourceAsStream(resourceName);
                InputStreamReader isr = new InputStreamReader(is, "UTF-8");
                BufferedReader reader = new BufferedReader(isr)) {

            Properties result = new Properties();

            // Properties#load() で渡す Reader オブジェクトを UTF-8 エンコーディング指定して生成した
            // InputStreamReader オブジェクトにする
            result.load(reader);

            return result;
        }
    }

    public static void main(String[] args) throws IOException {
        Properties Properties = loadUtf8Properties("/test.properties");
        System.out.println(Properties.getProperty("任意の日本語のキーを入力"));

    }
}
```

文字化けの理由

なぜ？文字化けが起きてしまうのか。。。

それは、 **文字コードの相違** で発生します。

例えば、日本語しか知らないのに、急にイタリア語で話かけられても対応できないですよね？
コンピューターにも読める文字、つまり **文字コード** が決まっています。
文字コードとは、 **人が扱う文字** を **コンピュータが理解できるような文字** に変換する時のフォーマットです。
そして日本語、英語、中国語・・・とあるように、文字コードにも200種類以上あります。

補足

今回の例で言うと、
`ResourceBundle bundle = ResourceBundle.getBundle("test");` でファイルを読み込み、
それを表現する時の文字コードが `ISO/IEC 8859-1` というものです。
ISO/IEC 8859-1では日本語(漢字)を表現することができません。
ゆえに、文字化けを防ぐには `UTF-8` 等に変換する必要があるという訳です。

ちなみにjava9からは、ResourceBundleのデフォルトの文字コードは **UTF-8** になります。
なので、今回のような煩わしさはなくなります。

課題

提出課題はありませんので、一通り学習が終わったら次の章に進んで下さい。

最終更新日時: 2022年 08月 21日(日曜日) 16:59