u58●直列化による転送モデル（EJBにおける分散オブジェクト）

JavaのクラスはSerializable（直列化インターフェース）をimplementsしたクラスを作成することで転送モデルが使えるようになります。

転送モデルとは、実体化したインスタンスを別のアプリケーションに転送できるということです。転送する手段としてファイルやネットワークを経由させることが可能です。

今回は、直列化クラスのインスタンスを一度ファイルに保存して、別のクラスから復元・表示してみたいと思います。

直列化クラスの作成

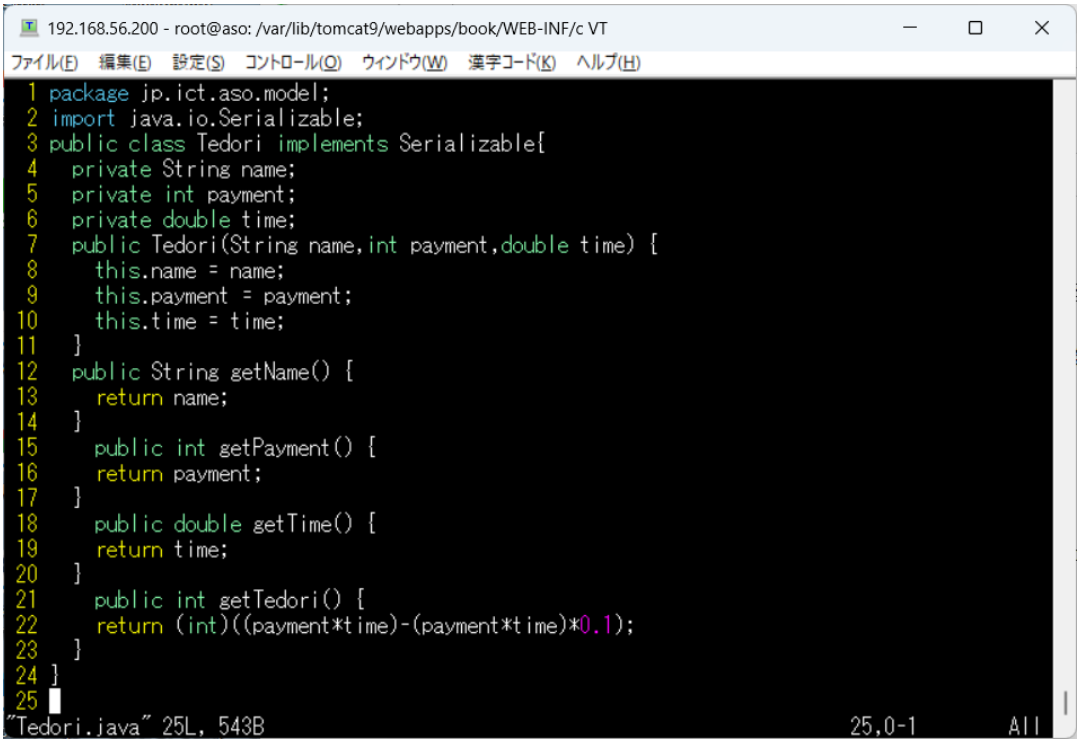
名前と時給と労働時間を設定すると10%の所得税を差し引いた手取り金額を計算するTedoriクラスを作ってみましょう。
このクラスは直列化インテーフェースを実装します。

以下作成場所は ～/libcon/WEB-INF/classes ですよ！

クラス図

```
------------------------------------  //クラス名
Tedori　(package:jp.ict.aso.model)
------------------------------------  //フィールド（プロパティ）
- name:String
- payment:int
- time:double
------------------------------------  //メソッド
+ Tedori(name:String,payment:int,time:double)
+ getName():String
+ getPayment():int
+ getTime():double
+ getTedori():int
------------------------------------
- private
+ public
```

Tedori.java




```java
package jp.ict.aso.model;
import java.io.Serializable;
public class Tedori implements Serializable{
  private String name;
  private int payment;
  private double time;
  public Tedori(String name,int payment,double time) {
    this.name = name;
    this.payment = payment;
    this.time = time;
  }
  public String getName() {
    return name;
  }
    public int getPayment() {
    return payment;
  }
    public double getTime() {
    return time;
  }
    public int getTedori() {
    return (int)((payment*time)-(payment*time)*0.1);
  }
}
```

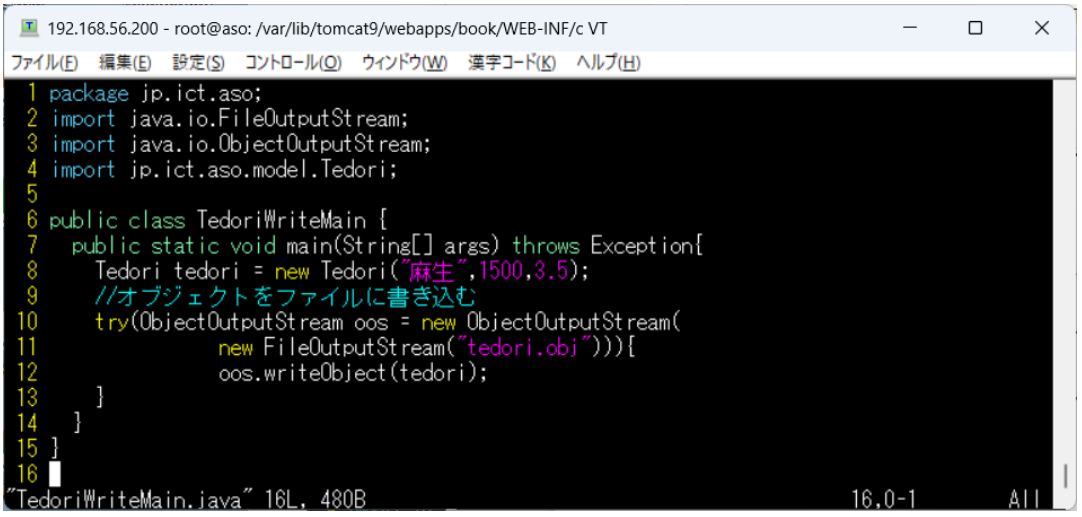
コンパイル

root@aso:classes# **javac -d . Tedori.java**

インスタンスの保存

Serializable実装クラスを実体化してインスタンス情報をファイルに書き込みます。
（インスタンス情報を直列化したバイナリデータをファイルに書き込みます）

TedoriWriteMain.java




```java
package jp.ict.aso;
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import jp.ict.aso.model.Tedori;

public class TedoriWriteMain {
  public static void main(String[] args) throws Exception{
    Tedori tedori = new Tedori("麻生",1500,3.5);
    //オブジェクトをファイルに書き込む
    try(ObjectOutputStream oos = new ObjectOutputStream(
              new FileOutputStream("tedori.obj"))){
              oos.writeObject(tedori);
    }
  }
}
```

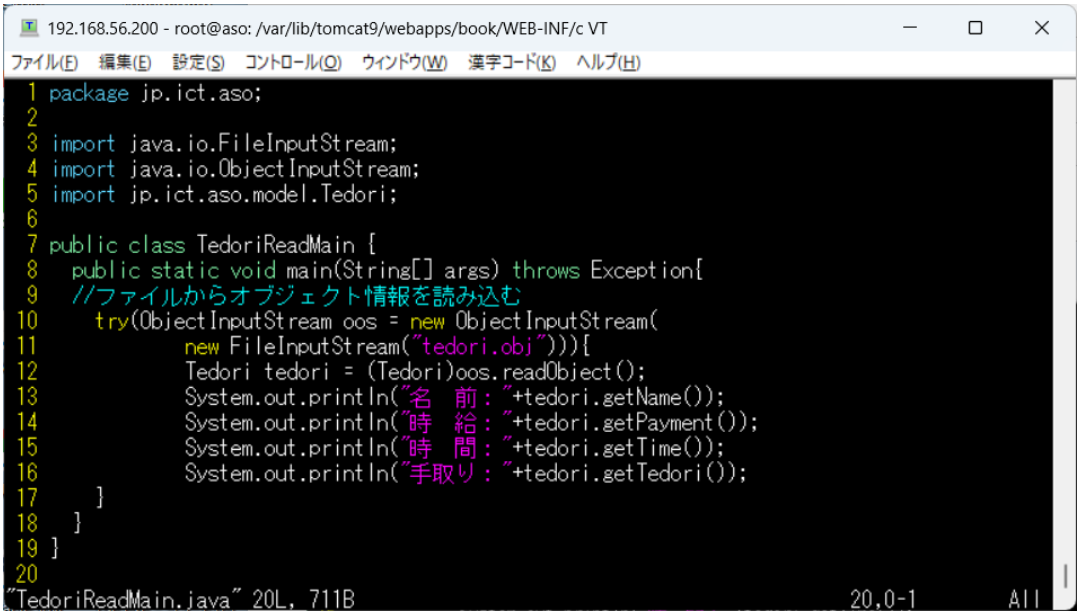
コンパイルと実行（実行することによりインスタンスが書き込まれます）

root@aso:classes# **javac -d . TedoriWriteMain.java**
root@aso:classes# **java jp.ict.aso.TedoriWriteMain**

インスタンスの読み出し

逆にバイナリデータからインスタンスを復元してみます。
（ファイルから直列化状態のインスタンスを読み込んで復元してみます）

TedoriReadMain.java



コンパイルと実行（実行することによりインスタンスが読み出されます）
root@aso:classes# **javac -d . TedoriReadMain.java**
root@aso:classes# **java jp.ict.aso.TedoriReadMain**
名　前：麻生
時　給：1500
時　間：3.5
手取り：4725

このようにSerializableなクラスは、「バイト直列化」が可能になります。
今回はファイルにインスタンスを保存・復元しましたが、インスタンスの「バイト直列化」が可能ということは、Stream経由でインスタンス情報をネットワークに送受信することも可能になります。
これから行うEnterpriseJavaBeansの分散オブジェクトもこの機能で実現しています。