THE UNIVERSITY COLLEGE
OF THE CARIBOO
Computing Science

COMP 113 - Computer Programming I (3,1,1)
Fall, 2004

Class Example – "Wallet"

Introduction

The following class, **wallet**, defines objects that function as very simple *virtual wallets***.** This means that simple *wallet*-type operations (methods) can be carried out, involving few value types (attributes).

Attributes:

- amount – the amount of money in the wallet; i.e., the balance
- ID_name – the wallet-holder's name*
- ID_address – address of the wallet-holder*
- pic – whether there are any pictures in the wallet*

** indicates that the attribute is not truly necessary, but included for the sake of representing real wallets*

Methods:

- wallet() – the default constructor; inits attributes to "zero"
- wallet() – the full constructor, with parameters for each attribute (ensures amount >= 0)
- addMoney() – increases the amount of the wallet, prevents negative value additions
- spendMoney() – deceases the amount of the wallet, prevents a negative amount
- getBalance() – [accessor] returns the value of the amount in the wallet (amount left unchanged)
- change() – [mutator] allows the *name, address,* and *picture status* values to change from parameter values
- show() – returns a String containing a formatted display of the current wallet's attributes
- equals() – returns a true/false, as to whether the current wallet "equals" another wallet in amount & name
- valid() – a helper method ensuring that a value is valid (>0), by returning the larger of the amount or zero
- main() – a tester main() allowing for an existing test to ensure the class is correct and reliable

The Class

```
/* File: wallet.java
   Author: Yanni Giftakis
   Purpose: this class defines a "virtual wallet" that can hold a certain
      amount (>=0), * along with some useful methods.
*/
public class wallet
{
    private double amount;      // the amount of money
    private String ID_name;     // name of wallet holder/owner
    private String ID_address;  // address of owner
    private boolean pic;        // any pictures

    //-------------------------------------------
    // default constructor; all values to "zero"
    public wallet()
    {
        amount = 0.0;       // no money
        ID_name = "";       // no owner
        ID_address = "";    // no address
        pic = false;        // no pictures
    }// end of wallet()
```

```
// full constructor; all values to parameters
public wallet(String iID_name, String iID_address, boolean ipic,
                  double iamount)
{
    ID_name = iID_name;         // owner
    ID_address = iID_address;   // address
    pic = ipic;                 // pictures
    amount = valid(iamount);    //  money
}// end of wallet()

//-------------------------------------------
// addMoney() - adds money to the wallet
public void addMoney (double value)
{
    amount = amount + valid(value);   // add valid money value to wallet
}// end of addMoney()

// spendMoney() - spends money in wallet
public void spendMoney (double value)
{
    amount = valid (amount - value);  // spends a valid amount from wallet
}// end of spendMoney()

// getBalance() - returns the current balance (amount)
public double getBalance()
{
    return (amount);
}// end of getBalance()

// change() - changes attributes, except for amount
public void change(String nID_name, String nID_address, boolean npic)
{
    ID_name = nID_name;         // owner
    ID_address = nID_address;   // address
    pic = npic;                 // pictures
}// end of change()

// show() - returns a string showing the attributes of the wallet
public String show()
{
    String output = "\nOwner is: "+ID_name
       + "\nAddress: "+ID_address
       + "\nPictures: "+pic
       + "\n, with an amount of $ "+amount;

    return (output);
}// end of show()

// equals() - returns if the other wallet equals the current wallet
public boolean equals (wallet other)
{
    boolean status;

    status = ( (amount == other.amount) &&
               (ID_name.equals(other.ID_name)) );

    return (status);
}// end of equals()
```

```
    //-----------------------------------------------
    // valid () - returns the valid (>=0) version of x
    private double valid (double x)
    {
        if (x < 0)    // if x is negative
           x = 0;         // set to zero

        return (x);   // return valid x value
    }// end of valid()


    //-----------------------------------------------
    // a tester main() is included to ensure the class works as expected
    public static void main(String[] args)
    {
        wallet Bobwallet = new wallet();  // def. constructor
        wallet Suewallet = new wallet("Sue","Kamloops",false,500.50);

        System.out.println("Bob:"+Bobwallet.show());
        System.out.println("Sue:"+Suewallet.show());

        Bobwallet.change ("Bob","Hawaii",true);
        System.out.println("Bob change:"+Bobwallet.show());

        Bobwallet.addMoney(200);
        System.out.println("Bob add 200:"+Bobwallet.show());
        System.out.println("Bob balance:"+Bobwallet.getBalance());

        Suewallet.spendMoney(200);
        System.out.println("Sue spend 200:"+Suewallet.show());

        Suewallet.spendMoney(3000);
        System.out.println("Sue spend 3000:"+Suewallet.show());

        System.out.println("Equality of wallets:"+Bobwallet.equals(Suewallet));

        wallet Newwallet = new wallet("Bob","Alaska",false,200);
        System.out.println("Equality of wallets:"+Bobwallet.equals(Newwallet));
    }// end of main()
}// end of class wallet
```