CST8253 Web Programming II

Lab 4

Objective

1. Work with C# Class and Objects

Due Date

See Brightspace posting for the due date of this lab. To earn 5 points, you are required:

1. Complete the lab as required.
2. Zip the project folder (the folder containing project file **.csproj**, C# source code **.cs** file) and submit the zipped file to the Brightspace before the due date.
3. Demo your lab work no later than the week following the lab's due date.

Requirements

Create a bank console application for opening and maintaining saving accounts for customers.

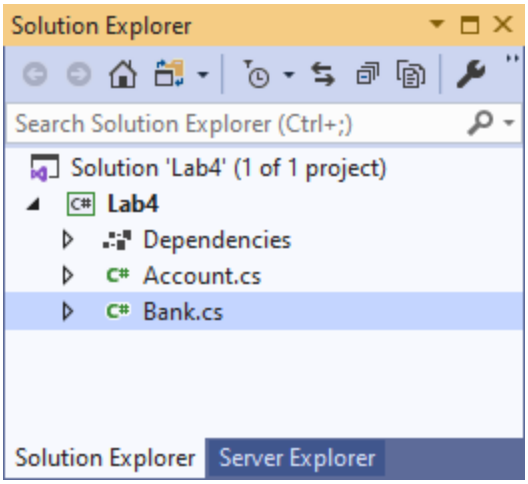
On start, the application will run according to the following steps:

1. Promot the user to enter the number of months the customer will keep the money in account.
2. To open an account for a new customer, the application first prompts the user to enter the customer's name.
3. It then prompts the user to enter an initial deposit amount for the saving account.
4. It then prompts the user to enter a monthly deposit amount to the saving account
5. It repeats step 2 – 4 to open a saving account for another customer until the user enters blank at step 1.
6. It then lists the balances of all accounts created in above steps after the number of months specified in the step 1, see below:

```
C:\OneDrive\OneDrive - Algonquin College\Algonquin\20W\CST8253\Week 4 - 5\Lab4\Lab4\bin\Debug...
Enter the number of months to deposit: 8

Enter Customer Name: Mary
Enter Mary's Initial Deposit Amount (minimum $1,000.00): 1200
Enter Mary's Monthly Deposit Amount (minimum $50.00): 60

Enter Customer Name: Peter
Enter Peter's Initial Deposit Amount (minimum $1,000.00): 1000
Enter Peter's Monthly Deposit Amount (minimum $50.00): 80

Enter Customer Name: John
Enter John's Initial Deposit Amount (minimum $1,000.00): 2000
Enter John's Monthly Deposit Amount (minimum $50.00): 100

Enter Customer Name:

After 8 month, Mary's account (#91473), has a balance of: $1,676.07
After 8 month, Peter's account (#91913), has a balance of: $1,633.44
After 8 month, John's account (#94595), has a balance of: $2,815.02

Press Enter to complete
```

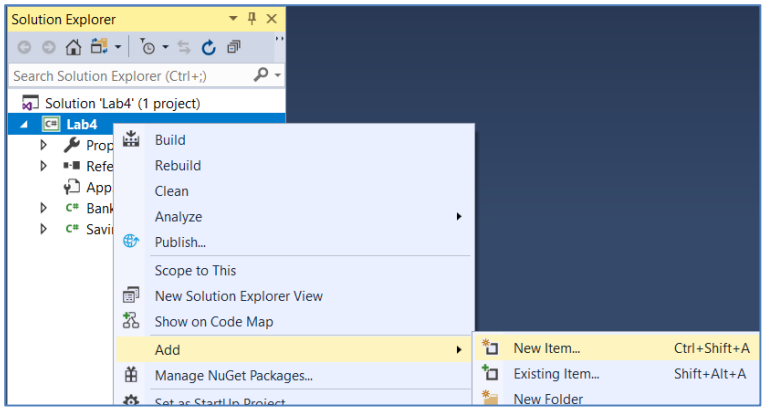
Implementation Notes

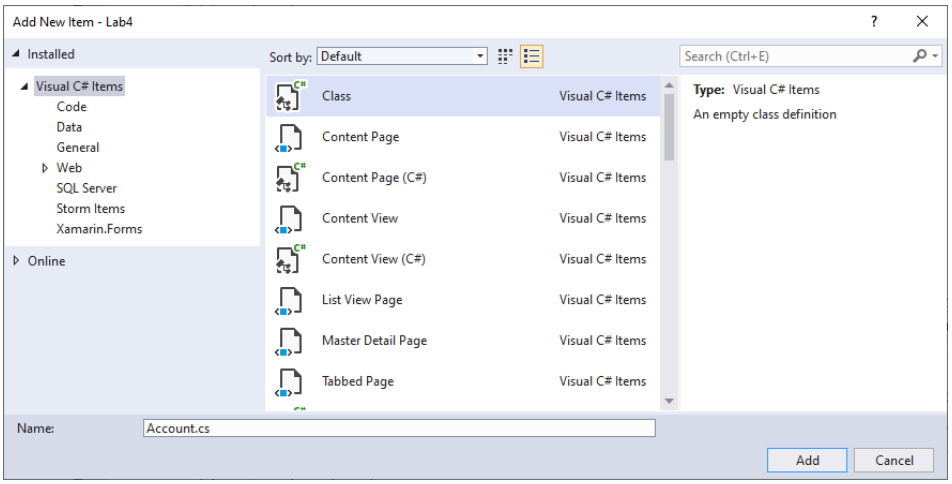
1. You solution should have two C# files; Bank.cs and Account.cs.



- Bank.cs contains the Main( ) method which is the starting point of the application's execution. You can rename the generated Program.cs to Bank.cs to start with.

- Account.cs contains the definition of the class Account. You can create this file by following the steps shown below:





2. The Account class should have **at least** the following properties

   - AccountNumber – An integer
   - OwnerName – the account owner's name
   - Balance – current balance of the account.
   - MonthlyDepositAmount – the amount that the customer will deposit into his/her account monthly.

3. The Account class also should have **at least** the following **static** properties:

   - MonthlyFee (initialize its value as 4.0)
   - MonthlyInterestRate (initialize its value as 0.0025)
   - MinimumInitialBalance (initialize its value as 1000)
   - MinimumMonthDeposit (initialize its value as 50)

4. The Account should have **at least** one constructor which takes the owner's name and initial deposit amount as parameter to initialize the properties of OwnerName and Balance.

The constructor also initializes the AccountNumber to a randomly generated integer of 9xxxx .

5. The Account should have **at least** two methods:

   Deposit – It takes a double as a parameter to increase the Balance.

   Withdraw – It takes a double as a parameter to decrease the Balance. In this lab, we will assume that the withdraw amount is always less than the current Balance amount.

6. For each month, the Bank class' Main method should update each account's balance in following order:

   - Deduct monthly fee
   - Add monthly interest
   - Add monthly deposit