Kenneth Robinson

MIS 320 - Assignment 2

February 1, 2011

# CONTROLS LIST

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **FORM** | **CONTROL** | **EVENT** | **PSUEDOCODE** |  |  |  |
| Login | lblName |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | lblAddress |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | lblEmail |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | lblProgramStatus |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | lblPhone |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | tbxName | lostFocus | if nothing is entered | |  |  |
|  |  |  |  | display error message | | |
|  |  |  |  |  |  |  |
|  | tbxAddress |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | tbxEmail | lostFocus | if text entered is not valid email | | |  |
|  |  |  |  | display error message | | |
|  |  |  |  |  |  |  |
|  | tbxPhone | lostFocus | if phone number is not 10 digits or 0 digits | | | |
|  |  |  |  | display error message | | |
|  |  |  |  |  |  |  |
|  | btnSubmit | click | if tbxName is blank | |  |  |
|  |  |  |  | display error message | | |
|  |  |  | if tbxPhone is in invalid format | | |  |
|  |  |  |  | display error message | | |
|  |  |  | if email is not valid | |  |  |
|  |  |  |  | display error message | | |
|  |  |  | if no errors |  |  |  |
|  |  |  |  | write form data to CSV file | | |
|  |  |  |  | if error writing form data | | |
|  |  |  |  |  | display error message | |
|  |  |  |  | else |  |  |
|  |  |  |  |  | clear fields | |
|  |  |  |  |  | display committed message | |
|  |  |  |  |  |  |  |
|  | btnReset | click | clear fields |  |  |  |
|  |  |  | update status message | |  |  |
|  |  |  |  |  |  |  |
|  | menubar |  |  |  |  |  |

# CLASS DESIGN

FormCustomerEntry

* Submit
* Check Name
* Check Email
* Check Phone
* Class Customer Info

CustomerDataAccess

* filePath
* customerNameDAO
* customerAddressDAO
* customerEmailDAO
* Committed
* Gets/sets
* Save Customer

CustomerInfo

* customerName
* customerAddress
* customerEmail
* customerPhone
* committed
* errorInfo
* gets/sets
* save Customer
* class CustomerDataAccess

# CLASS DEFINITIONS

1. Class FormCustomerEntry (User Interface layer)
   1. Description
      1. User Interface layer; does error checking on form
   2. Members
      1. none
   3. sub btnSubmit\_Click
      1. create new instances of Customer Information class
      2. Set variables in formCustomerEntry to variables in CustomerInformation class
      3. Check name field
      4. Check email field
      5. Check phone field
      6. If there are no errors with name, email, and phone field, then call CustomerInformation.SaveCust
      7. Get CustomerInformation.custErrorIfno
      8. If the commit was not successful then display error message; otherwise display saved message and clear fields
   4. Sub ExitTooStripMenuItem\_Click
      1. Exit program
   5. Sub btnReset\_click
      1. Update status
      2. Clear all fields
   6. tbxPhone\_textChange loseFocus
      1. check phone number
      2. update status if there is an error with phone number
   7. updatestatus
      1. if there is an error place “ERROR!” in front of message; otherwise display status message
   8. checkphone
      1. get the length of the phone number; if too short or too long or not numeric, then display error message
   9. checkname
      1. if the field is empty (null) then display error message
   10. checkEmail
       1. if the length of the text is 0 then don’t check
          1. if the text entered is in the same format as email string, then continue; else display error message
   11. tbxName\_textChange LoseFocus
       1. check name
       2. update status if there is an error with name
   12. tbxEmail\_textChanged LoseFocus
       1. check email
       2. update status if there is an error with name
2. class CustomerInfo (Business Logic layer)
   1. Definition
      1. Gets customer info from the UI moves it to the DA
   2. Members
      1. CustomerName
      2. CustomerAddress
      3. CustomerEmail
      4. CustomerPhone
      5. Committed
      6. errorInfo
   3. Methods
      1. saveCust
         1. set variables in CustomerInfo class to CustomerDAO class
         2. call CustomerDAO.saveCustDAO
         3. if the commit was not successful, display error message; otherwise display saved message
3. Class DataAccess (Data Access Layer)
   1. Definition
      1. Saves the customer info in the BL to a CSV file.
   2. Members
      1. filePath
      2. customerNameDAO
      3. customerAddressDAO
      4. customerEmailDAO
      5. customerPhoneDAO
      6. committed
   3. Methods
      1. saveCustDAO
         1. check if file already exists
         2. try to create new file (StreamWriter)
            1. if the file doesn’t exist write the header; then write the form information to the text file.
            2. Change the committed status
            3. Close the file
         3. If there was an exception, throw the message and report commitment error