```
/*-----------------------------------------------------------------------------------------
 * Name:        Dan Cassidy
 * Date:        2015-06-09
 * Assignment:  cView-P2
 * Source File: CViewDataInteractive.cs
 * Course:      CSCI-C 490, C# Programming, MoWe 08:00
 * Purpose:     Provides interactive management of a CViewDataSet object.
-----------------------------------------------------------------------------------------*/

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CView
{
    class CViewDataInteractive
    {
        //Helper constants for menu validation.
        private const mainMenu MAINMENU_MIN = mainMenu.ADD;
        private const mainMenu MAINMENU_MAX = mainMenu.EXIT;
        private const fieldMenu FIELDMENU_MIN = fieldMenu.NAME;
        private const fieldMenu FIELDMENU_MAX = fieldMenu.BACK;

        //Primary class field/instance variable.
        private CViewDataSet data = new CViewDataSet();

        //Enum for the main menu. Basic code idea from Stack Overflow.
        //http://stackoverflow.com/a/15752719
        private enum mainMenu
        {
            ADD = 1,
            MODIFY,
            SEARCH,
            DELETE,
            DISPLAY_ALL,
            EXIT
        }

        //Enum for the modify menu. Basic code idea from Stack Overflow.
        //http://stackoverflow.com/a/15752719
        private enum fieldMenu
        {
            NAME = CViewData.FIELDS_MIN,
            FACILITY,
            ADDRESS,
            CITY,
            PHONE,
            BACK
        }

        /*-----------------------------------------------------------------------------------
         * Method:  InteractiveManipulation
         * Purpose: Entry point for interactive manipulation of CViewDataSet object.
         * Input:   Nothing.
         * Output:  Nothing.
        -----------------------------------------------------------------------------------*/
        public void InteractiveManipulation()
        {
            //Loop the main menu until the user decides to exit.
            while (MainMenuAction(MainMenuDisplay()) != mainMenu.EXIT) ;
        }

        /*------------------------------------------------------------------------------------------
         * Method:  DataAdd
         * Purpose: Interactively add an item based on the user's input.
         * Input:   Nothing.
         * Output:  Nothing.
        ------------------------------------------------------------------------------------------*/
        private void DataAdd()
        {
            //New CViewData object that will be added to the dataset.
            CViewData dataToAdd = new CViewData();

            //Prompt the user to input information about the new item.
            Console.WriteLine("----------------");
            Console.WriteLine("| Add New Item |");
            Console.WriteLine("----------------");
            Console.Write("Facility Name: ");
            dataToAdd.Name = Console.ReadLine();
            Console.Write("Facility Type: ");
            dataToAdd.FacilityType = Console.ReadLine();
            Console.Write("Address: ");
            dataToAdd.Address = Console.ReadLine();
            Console.Write("City: ");
            dataToAdd.City = Console.ReadLine();
            Console.Write("Phone Number: ");
            dataToAdd.PhoneNumber = Console.ReadLine();

            //Extra line for formatting.
            Console.WriteLine();

            //Add the new item to the main data set.
            data.Add(dataToAdd);

            //Sort the data set.
            data.SortByName();
        }

        /*------------------------------------------------------------------------------------------
         * Method:  DataDelete
         * Purpose: Interactively deletes an object based upon user input.
         * Input:   Nothing.
         * Output:  Nothing.
        ------------------------------------------------------------------------------------------*/
        private void DataDelete()
        {
            //Default value of 0 in case the user doesn't enter a choice and just hits 'enter'.
            int indexToDelete = 0;

            //Display the user's choice.
            Console.WriteLine("----------------------------------");
            Console.WriteLine("| Delete Item -- Existing Items |");
            Console.WriteLine("----------------------------------");

            //Display a numbered list of all the objects in the data set.
            DataDisplayAllNumbered();

            //Get the user's choice of which object to delete.
            Console.Write("\nSelect item (0 to cancel): ");
            int.TryParse(Console.ReadLine(), out indexToDelete);
            indexToDelete--;

            //Extra line for formatting.
            Console.WriteLine();

            //Display the results.
            Console.WriteLine("-------------------------");
            Console.WriteLine("| Delete Item -- Results |");
            Console.WriteLine("-------------------------");

            //Validate the user's choice.
            if (indexToDelete == -1)
            {
                //The user changed their mind.
                Console.WriteLine("Cancelled.\n");
                return;
            }
            else if (indexToDelete < 0 || indexToDelete >= data.Count)
            {
                //The user input an invalid object index.
                Console.WriteLine("Invalid item.\n");
                return;
            }

            //Delete the object and display confirmation of its deletion.
            data.Delete(indexToDelete);
            Console.WriteLine("Item {0} has been deleted.\n", indexToDelete + 1);

            //Display the still existing items.
            DataDisplayAll(false);
        }

        /*---------------------------------------------------------------------------------------
         * Method:  DataDisplayAll
         * Purpose: Displays the header and the serialized dataset object.
         * Input:   bool displayTitle, determines whether the method should print a title showing
         *          that this method was the one that was called.
         * Output:  Nothing.
        ---------------------------------------------------------------------------------------*/
        private void DataDisplayAll(bool displayTitle = true)
        {
            //Choose whether to display the title.
            if (displayTitle)
            {
                //Display the user's choice.
                Console.WriteLine("---------------------");
                Console.WriteLine("| Display All Items |");
                Console.WriteLine("---------------------");
            }

            //Display all the objects.
            Console.WriteLine("{0}\n{1}", data.Header, data.Count != 0 ?
                              data.ToString() : "No items currently stored.\n");
        }

        /*---------------------------------------------------------------------------------------
         * Method:  DataDisplayAllNumbered
         * Purpose: Display a header and a numbered list of objects.
         * Input:   Nothing.
         * Output:  Nothing.
        ---------------------------------------------------------------------------------------*/
        private void DataDisplayAllNumbered()
        {
            //Display the header.
            Console.WriteLine("Item  {0}", data.Header);

            //If the dataset is not empty.
            if (data.Count != 0)
                //Display the numbered objects, starting at 1.
                for (int objectNum = 0; objectNum < data.Count; objectNum++)
```

```
                    Console.WriteLine("{0,4}  {1}", objectNum + 1, data[objectNum]);
            else
                //Display a message saying that dataset is empty.
                Console.WriteLine("No items currently stored.");
        }

        /*--------------------------------------------------------------------------------------------
         * Method:  DataModify
         * Purpose: Interactively modifies an object based on the user's input.
         * Input:   Nothing.
         * Output:  Nothing.
        --------------------------------------------------------------------------------------------*/
        private void DataModify()
        {
            //Default value of 0 in case the user doesn't enter a choice and just hits 'enter'.
            int indexToModify = 0;

            //Display the user's choice.
            Console.WriteLine("---------------------------------");
            Console.WriteLine("| Modify Item -- Existing Items |");
            Console.WriteLine("---------------------------------");

            //Display a numbered list of all the objects in the data set.
            DataDisplayAllNumbered();

            //Get the user's choice of which object to delete.
            Console.Write("\nSelect item (0 to cancel): ");
            int.TryParse(Console.ReadLine(), out indexToModify);
            indexToModify--;

            //Extra line for formatting.
            Console.WriteLine();

            //Validate the user's choice.
            if (indexToModify == -1)
            {
                //The user changed their mind.
                Console.WriteLine("Cancelled.\n");
                return;
            }
            else if (indexToModify < 0 || indexToModify >= data.Count)
            {
                //The user input an invalid object index.
                Console.WriteLine("Invalid item.\n");
                return;
            }

            do
            {
                //Display the chosen object.
                Console.WriteLine("-------------------------------");
                Console.WriteLine("| Modify Item -- Chosen Item |");
                Console.WriteLine("-------------------------------");
                Console.WriteLine("{0}\n{1}\n", data.Header, data[indexToModify]);

                //Loop while the use has not chosen to go back.
            } while (DataModifyMenuAction(FieldMenuDisplay(), indexToModify) != fieldMenu.BACK);
        }

        /*--------------------------------------------------------------------------------------------
         * Method:  DataModifyMenuAction
         * Purpose: Acts on the user's choice made at the Modify Menu.
         * Input:   fieldMenu choice, represents the action specified.
         * Output:  fieldMenu, represents the action specified.
        -----------------------------------------------------------------------------------------*/
        private fieldMenu DataModifyMenuAction(fieldMenu choice, int indexToModify)
        {
            //Decide what to do based on the user's choice.
            switch (choice)
            {
                case fieldMenu.NAME:
                    //Change the name of the item.
                    Console.WriteLine("Current Facility Name: {0}", data[indexToModify].Name);
                    Console.Write("New Facility Name: ");
                    data[indexToModify].Name = Console.ReadLine();

                    //Sort the data set after changing the name since name is the sort criteria.
                    data.SortByName();

                    break;

                case fieldMenu.FACILITY:
                    //Change the facility type of the item.
                    Console.WriteLine("Current Facility Type: {0}",
                                      data[indexToModify].FacilityType);
                    Console.Write("New Facility Type: ");
                    data[indexToModify].FacilityType = Console.ReadLine();

                    break;

                case fieldMenu.ADDRESS:
                    //Change the address of the item.
                    Console.WriteLine("Current Address: {0}", data[indexToModify].Address);
                    Console.Write("New Address: ");
                    data[indexToModify].Address = Console.ReadLine();

                    break;

                case fieldMenu.CITY:
                    //Change the city of the item.
                    Console.WriteLine("Current City: {0}", data[indexToModify].City);
                    Console.Write("New City: ");
                    data[indexToModify].City = Console.ReadLine();

                    break;

                case fieldMenu.PHONE:
                    //Change the phone number of the item.
                    Console.WriteLine("Current Phone Number: {0}", data[indexToModify].PhoneNumber);
                    Console.Write("New Phone Number: ");
                    data[indexToModify].PhoneNumber = Console.ReadLine();

                    break;

                case fieldMenu.BACK:
                    //Nothing to do; the user wants to go back.
                default:
                    //Catch-all.
                    return choice;
            }

            //Extra line for formatting.
            Console.WriteLine();

            //Return choice so the calling method knows what the choice was and can act accordingly.
            return choice;
        }

        /*------------------------------------------------------------------------------------------
         * Method:  DataSearch
         * Purpose: Interactively searches for objects based upon user input.
         * Input:   Nothing.
         * Output:  Nothing.
        ----------------------------------------------------------------------------------------*/
        private void DataSearch()
        {
            do
            {
                //Display the user's choice.
                Console.WriteLine("----------------");
                Console.WriteLine("| Search Items |");
                Console.WriteLine("----------------");

                //Loop while the use has not chosen to go back.
            } while (DataSearchMenuAction(FieldMenuDisplay()) != fieldMenu.BACK);
        }

        /*------------------------------------------------------------------------------------------
         * Method:  DataSearchMenuAction
         * Purpose: Acts on the user's choice made at the Search Menu.
         * Input:   fieldMenu choice, represents the action specified.
         * Output:  fieldMenu, represents the action specified.
        ----------------------------------------------------------------------------------------*/
        private fieldMenu DataSearchMenuAction(fieldMenu choice)
        {
            //Decide what to display based on the user's choice.
            switch (choice)
            {
                case fieldMenu.NAME:
                    //Search the name field.
                    Console.WriteLine("---------------------------------");
                    Console.WriteLine("| Search Items -- Facility Name |");
                    Console.WriteLine("---------------------------------");
                    break;

                case fieldMenu.FACILITY:
                    //Search the facility type field.
                    Console.WriteLine("---------------------------------");
                    Console.WriteLine("| Search Items -- Facility Type |");
                    Console.WriteLine("---------------------------------");
                    break;

                case fieldMenu.ADDRESS:
                    //Search the address field.
                    Console.WriteLine("---------------------------");
                    Console.WriteLine("| Search Items -- Address |");
                    Console.WriteLine("---------------------------");
                    break;

                case fieldMenu.CITY:
                    //Search the city field.
                    Console.WriteLine("------------------------");
                    Console.WriteLine("| Search Items -- City |");
                    Console.WriteLine("------------------------");
                    break;

                case fieldMenu.PHONE:
                    //Search the phone number field.
                    Console.WriteLine("--------------------------------");
                    Console.WriteLine("| Search Items -- Phone Number |");
                    Console.WriteLine("--------------------------------");
                    break;

                case fieldMenu.BACK:
                    //Nothing to do; the user wants to go back.
                default:
                    //Catch-all.
                    return choice;
            }

            //Ask for the search text.
            Console.Write("Enter your search text: ");

            //Get the user's search text and pipe that directly into the search method.
            CViewDataSet foundData = data.Search(Console.ReadLine(), (CViewData.Fields)choice);

            //Show the number of items found.
            Console.WriteLine("{0} item{1} found.\n", foundData.Count,
                              foundData.Count == 1 ? "" : "s");

            //If any items found, display them.
            if (foundData.Count != 0)
                Console.WriteLine("{0}\n{1}", foundData.Header, foundData);

            //Return choice so the calling method knows what the choice was and can act accordingly.
            return choice;
        }

        /*---------------------------------------------------------------------------------------
         * Method:  FieldMenuDisplay
         * Purpose: Display the field menu and get a choice. Must have valid input to return.
         * Input:   Nothing.
         * Output:  fieldMenu, representing the choice that was made.
        ---------------------------------------------------------------------------------------*/
        private fieldMenu FieldMenuDisplay()
        {
            fieldMenu menuChoice = 0;
            bool invalid = true;

            do
            {
                //Display the menu.
                Console.WriteLine("Please select the field you would like to work with:");
                Console.WriteLine("  1) Facility Name");
                Console.WriteLine("  2) Facility Type");
                Console.WriteLine("  3) Street Address");
                Console.WriteLine("  4) City");
                Console.WriteLine("  5) Phone Number");
                Console.WriteLine("  6) Back");
                Console.Write("Choice: ");

                //Get the user's choice.
                string input = Console.ReadLine();

                //Extra line for formatting.
                Console.WriteLine();

                //Validate the user input.
                invalid = !fieldMenu.TryParse(input, out menuChoice) ||
                          !FieldMenuValidate(menuChoice);
            } while (invalid);

            //Return the user's choice.
            return menuChoice;
        }

        /*-------------------------------------------------------------------------------------------
         * Method:  FieldMenuValidate
         * Purpose: Validates that the choice by the user is within the limits and is logically
         *          possible.
         * Input:   mmodifyMenu value, contains the user's choice.
         * Output:  bool, representing whether the user's choice was valid or not.
        -------------------------------------------------------------------------------------------*/
        private bool FieldMenuValidate(fieldMenu value)
        {
            //Check to make sure that the user input is within valid limits.
            if (value < FIELDMENU_MIN || value > FIELDMENU_MAX)
                return false;

            //Otherwise, input is good.
            return true;
        }

        /*-------------------------------------------------------------------------------------------
         * Method:  MainMenuAction
         * Purpose: Acts on the user's choice made at the Main Menu.
         * Input:   mainMenu choice, represents the action specified.
         * Output:  mainMenu, represents the action specified.
        -------------------------------------------------------------------------------------------*/
        private mainMenu MainMenuAction(mainMenu choice)
        {
            //Decide what to do based on the user's choice.
            switch (choice)
            {
                case mainMenu.ADD:
                    //Add a new item.
                    DataAdd();
                    break;

                case mainMenu.MODIFY:
                    //Modify an existing item.
                    DataModify();
                    break;

                case mainMenu.SEARCH:
                    //Search items.
                    DataSearch();
                    break;

                case mainMenu.DELETE:
                    //Delete an item.
                    DataDelete();
                    break;

                case mainMenu.DISPLAY_ALL:
                    //Display all the items.
                    DataDisplayAll();
                    break;

                case mainMenu.EXIT:
                    //Do nothing, exiting the method.
                default:
                    //Catch-all.
                    break;
            }

            //Return choice so the calling method knows what the choice was and can act accordingly.
            return choice;
        }

        /*-------------------------------------------------------------------------------------------
         * Method:  MainMenuDisplay
         * Purpose: Display the main menu and get a choice. Must have valid input to return.
         * Input:   Nothing.
         * Output:  mainMenu, representing the choice that was made.
        -------------------------------------------------------------------------------------------*/
        private mainMenu MainMenuDisplay()
        {
            mainMenu menuChoice = 0;
            bool invalid = true;

            do
            {
                //Display the menu.
                Console.WriteLine("------------------------");
                Console.WriteLine("| Main Interactive Menu |");
                Console.WriteLine("------------------------");
                Console.WriteLine("Please select an option:");
                Console.WriteLine("  1) Add New Item");
                Console.WriteLine("  2) Modify Item");
                Console.WriteLine("  3) Search Items");
                Console.WriteLine("  4) Delete Item");
                Console.WriteLine("  5) Display All Items");
                Console.WriteLine("  6) Exit");
                Console.Write("Choice: ");

                //Get the user's choice.
                string input = Console.ReadLine();

                //Extra line for formatting.
                Console.WriteLine();

                //Validate the user input.
                invalid = !mainMenu.TryParse(input, out menuChoice) ||
                          !MainMenuValidate(menuChoice);
            } while (invalid);

            //Return the user's choice.
            return menuChoice;
        }

        /*-------------------------------------------------------------------------------------------
         * Method:  MainMenuValidate
         * Purpose: Validates that the choice by the user is within the limits and is logically
         *          possible.
         * Input:   mainMenu value, contains the user's choice.
         * Output:  bool, representing whether the user's choice was valid or not.
        -------------------------------------------------------------------------------------------*/
        private bool MainMenuValidate(mainMenu value)
        {
            //Check to make sure that the user input is within valid limits.
            if (value < MAINMENU_MIN || value > MAINMENU_MAX)
                return false;

            //If the data set is empty, limit user to adding an entry or exiting.
            if (data.Count == 0 && (value != mainMenu.ADD && value != mainMenu.EXIT))
            {
                Console.WriteLine("No data is present. Please choose a different option.\n");
                return false;
            }

            //Otherwise, input is good.
            return true;
        }
```

```
    }
}

```