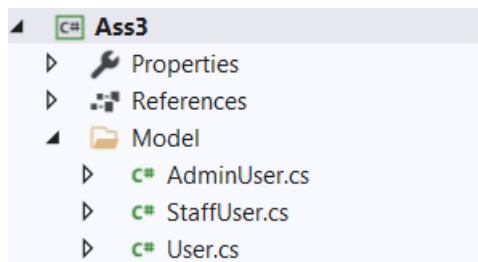


Assignment #3

Code Report :

First I have add model name space and add 3 classes inside it:



1. User class which has this attributes :

- a. Id
- b. Name
- c. Age

And functions:

- a. User constructor with (int id , string name, int age) to set values of its attributes.
- b. Override to string function to print user id , name, age values.

- c. Override equals function to compare the object based on its type and their data.

```
public class User
{
    11 references
    public int id { set; get; }
    23 references
    public String name { set; get; }
    23 references
    public int age { set; get; }

    4 references
    public User(int id , String name , int age)
    {
        this.id = id;
        this.name = name;
        this.age = age;
    }

    5 references
    public override string ToString()
    {
        return $" - Id: {this.id}\n - Name: {this.name}\n - Age: {this.age}";
    }

    2 references
    public override bool Equals(object value)
    {
        if ((ReferenceEquals(null, value)) || (value.GetType() != this.GetType())) return false;
        if (ReferenceEquals(this, value)) return true;

        User user = value as User;

        return (user != null)
            && (id == user.id)
            && (name == user.name)
            && (age == user.age);
    }

    2 references
    public override int GetHashCode()
    {
        return base.GetHashCode();
    }
}
```

2. Admin User class which inherits from user and has this attribute :

- a. List<StaffUser>

And functions:

- d. AdminUser constructor with (int id , string name, int age) to passed them to User "parent " constructor and implement it.
- e. Override to string function to print Admin user id , name, age values and staff list values by call parent to string override function and staff user to string over ride function .
- f. Override equals function to compare the object based on its type and their data.

```

16 references
internal class AdminUser : User
{
    public List<StaffUser> staffUser = null;

    2 references
    public AdminUser(int id, String name, int age) : base(id, name, age)
    {
        this.staffUser = new List<StaffUser>();
    }

    3 references
    public override string ToString()
    {
        string admin = base.ToString()+"\n - Staff User :\n";

        string staffUser_ = "";

        for(int i = 0; i < staffUser.Count; i++)
        {
            staffUser_ += staffUser[i].ToString() + "\n";
        }

        return admin + staffUser_ ;
    }

    1 reference
    public override bool Equals(object value)
    {
        if ((ReferenceEquals(null, value)) || (value.GetType() != this.GetType())) return false;
        if (ReferenceEquals(this, value)) return true;

        AdminUser user = value as AdminUser;

        return (user != null)
            && (id == user.id)
            && (name == user.name)
            && (age == user.age)
            && (staffUser == user.staffUser);
    }

    3 references
    public override int GetHashCode()
    {
        return base.GetHashCode();
    }
}

```

✓ No issues found

3. Staff User class which inherits from user and has this attribute :

a. Enum Role role;

And functions:

g. User constructor with (int id , string name, int age, Role role) parameter to set values of its attributes.

h. Override to string function to print user id , name, age values by call parent to string override function and print role .

i. Override equals function to compare the object based on its type and their.

```
18 references
public class StaffUser : User
{
    internal Role role;

    2 references
    public StaffUser(int id, String name, int age, Role role) : base(id, name, age)
    {
        this.role = role;
    }

    4 references
    public override string ToString()
    {
        return base.ToString() + $"\\n - Role: {role}";
    }

    1 reference
    public override bool Equals(object value)
    {
        if ((ReferenceEquals(null, value)) || (value.GetType() != this.GetType())) return false;
        if (ReferenceEquals(this, value)) return true;

        StaffUser user = value as StaffUser;

        return (user != null)
            && (id == user.id)
            && (name == user.name)
            && (age == user.age)
            && (role == user.role);
    }

    3 references
    public override int GetHashCode()
    {
        return base.GetHashCode();
    }
}

7 references
public enum Role
{
    Role1 ,
    Role2,
    Role3
}
```

✓ No issues found

Second thing I have create ModelView Class :

In Model View class, users can add, edit, and delete User, StaffUser and AdminUser objects.

1. I have initialize 3 ObservableCollection one for Users , another for Admin users , third for Staff users . and set their value on ModelView constructor.

```

public ObservableCollection<User> users = null;
public ObservableCollection<StaffUser> staffUsers = null;
public ObservableCollection<AdminUser> adminUsers = null;
public delegate void OnReciveEvent(object sender, EventArgs args);
public OnReciveEvent onReciveEvent;
1 reference
public ModelView()
{
    users = new ObservableCollection<User>();
    staffUsers = new ObservableCollection<StaffUser>();
    adminUsers = new ObservableCollection<AdminUser>();
}

```

2. I have create AddUser function which receive user type and object from user or staff user or admin user and make add the user to one of observable collection according to user type.

```

3 references
public string AddUser<T>(int userType,T user)
{
    if (userType == 1)
    {
        users.Add(user as User);
    }else if (userType == 2)
    {
        staffUsers.Add(user as StaffUser);
    }else if(userType == 3)
    {
        adminUsers.Add(user as AdminUser);
    }

    if (onReciveEvent != null)
    {
        onReciveEvent(this,EventArgs.Empty);
    }

    return "User has add successfully : ";
}

```

3. I have create EditUser function which receive user type , object from user or staff user or admin user and user index to edit them and make editing on this user in its observable list .

3 references

```
public string EditUser<T>(int userType, T user, int userIndex)
{
    if (userType == 1)
    {
        User user_ = user as User;

        if (String.IsNullOrEmpty(user_.name))
        {
            users[userIndex].name = users[userIndex].name;
        } else
        {
            users[userIndex].name = user_.name;
        }

        if (user_.age == 0)
        {
            users[userIndex].age = users[userIndex].age;
        } else
        {
            users[userIndex].age = user_.age;
        }
    }
    else if (userType == 2)
    {
        StaffUser user_ = user as StaffUser;

        if (String.IsNullOrEmpty(user_.name))
        {
            staffUsers[userIndex].name = staffUsers[userIndex].name;
        }
    }
}
```

4. I have create RemoveUser function which receive user type and user index in observable list you want remove it and make deleting for user from observable list according user type .

3 references

```
public string RemoveUser(int userType, int userIndex)
{
    if (userType ==1)
    {
        users.Remove(users[userIndex]);
    }
    else if (userType == 2)
    {
        staffUsers.Remove(staffUsers[userIndex]);
    }
    else if (userType == 3)
    {
        adminUsers.Remove(adminUsers[userIndex]);
    }

    return "User has deleted successfully : ";
}
```

5. I have create GetUserIndex function which receive user type and user id to return index of user in observable list you want according user type you have choose .

12 references

```
public int GetUserIndex(int userType, int userID)
{
    int indexOfUser = -1;
    if (userType == 1)
    {
        for (int i = 0; i < users.Count; i++)
        {
            if (users[i].id == userID)
            {
                indexOfUser = i;
            }
        }
    }
    if (userType == 2)
    {
        for (int i = 0; i < staffUsers.Count; i++)
        {
            if (staffUsers[i].id == userID)
            {
                indexOfUser = i;
                return indexOfUser;
            }
        }
    }
    if (userType == 3)
    {
        for (int i = 0; i < adminUsers.Count; i++)
        {
            if (adminUsers[i].id == userID)
            {
                indexOfUser = i;
                return indexOfUser;
            }
        }
    }
}
```

6. Finally I have create `UserIsFound` that return `NotFoundedIDException` exception if user dos not founded and true if the user is found.

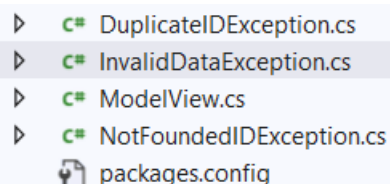
8 references

```
public bool UserIsFound(int userType, int userID)
{
    if (GetUserIndex(userType, userID) < 0)
        throw new NotFoundedIDException("User tries to update/delete/add a user that is not found.");
    else
        return true;
}
```

Second thing I have create View Class :

7. I have create implementation function that's will display menu for user to choose the type of user collection he/she want make add/edit/delet data from it , after choose user type you will choose the type of operation you want to do add/edit/delet then according to your choice the program will call the functions from ModelView Simultaneously with this process, I will throw my exception that I have defined them in enter values operation and call

model views functions , my exceptions are :



```
▶ C# DuplicateIDException.cs
▶ C# InvalidDataException.cs
▶ C# ModelView.cs
▶ C# NotFoundedIDException.cs
📄 packages.config
```

1. `DuplicateIDException` : that's will return message for user said you are tries to add a new user with a duplicate id.
like if the user want to add a new user first he/she will enter user id then chick if the id is exist in observable list at previous time then the program will throw a `DuplicateIDException` exception .
2. `InvalidDataException` : that's will return message for user said you are enters invalid data.
like if the user want to add a new user he/she must set id,name and age and then the program will throw a `DuplicateIDException` exception if id not int or name not string or age not int .

3. `NotFoundedIDException`: that's will return message for user said you are tries to update/delete a user that is not found.

like if the user want to delete or edit a specific user in a specific observable collection , he/she must set id and then the program will throw a `NotFoundedIDException` exception if id not found in observable collection .

for example :

```
8 references
public bool UserIsFound(int userType, int userID)
{
    if (GetUserIndex(userType, userID) < 0)
        throw new NotFoundedIDException("User tries to update/delete/add a user that is not found.");
    else
        return true;
}
```

Finally I have delegate to create an event handler to listion to any

change mode on the model classes and print the change in the screen

.

```
ModelView.users.CollectionChanged += ConllectionAddChanged<User>;
ModelView.users.CollectionChanged += ConllectionRemoveChanged<User>;

ModelView.staffUsers.CollectionChanged += ConllectionAddChanged<StaffUser>;
ModelView.staffUsers.CollectionChanged += ConllectionRemoveChanged<StaffUser>;

ModelView.adminUsers.CollectionChanged += ConllectionAddChanged<AdminUser>;
ModelView.adminUsers.CollectionChanged += ConllectionRemoveChanged<AdminUser>;
```

And add `ConllectionAddChanged` & `ConllectionRemoveChanged`

function to each one that will chick the event and print the changes

at the screen like all users data and users count :

3 references

```
private static void ConllectionAddChanged<T>(object sender, NotifyCollectionChangedEventArgs e)
{
    if (e.Action.Equals(NotifyCollectionChangedAction.Add))
    {
        ObservableCollection<T> user = (ObservableCollection<T>)sender;
        Console.WriteLine($"** Total count After Add is : {user.Count}");
        Console.WriteLine($"** The urrent user in the system are :\n");
        for (int i=0; i < user.Count;i++)
        {
            Console.WriteLine(user[i].ToString());
            Console.WriteLine(".....");
        }
    }
}
```

3 references

```
private static void ConllectionRemoveChanged<T>(object sender, NotifyCollectionChangedEventArgs e)
{
    if (e.Action.Equals(NotifyCollectionChangedAction.Remove))
    {
        ObservableCollection<T> user = (ObservableCollection<T>)sender;
        Console.WriteLine($"** Total count after Remove is : {user.Count}");
        Console.WriteLine($"** The urrent user in the system are :\n");
        for (int i = 0; i < user.Count; i++)
        {
            Console.WriteLine(user[i].ToString());
            Console.WriteLine(".....");
        }
    }
    else if (e.Action.Equals(NotifyCollectionChangedAction.Reset))
    {
        ObservableCollection<T> user = (ObservableCollection<T>)sender;
        Console.WriteLine($"**Total count after Reset is : {user.Count}");
    }
}
```

Done 😊

