Module 2 – To Do List

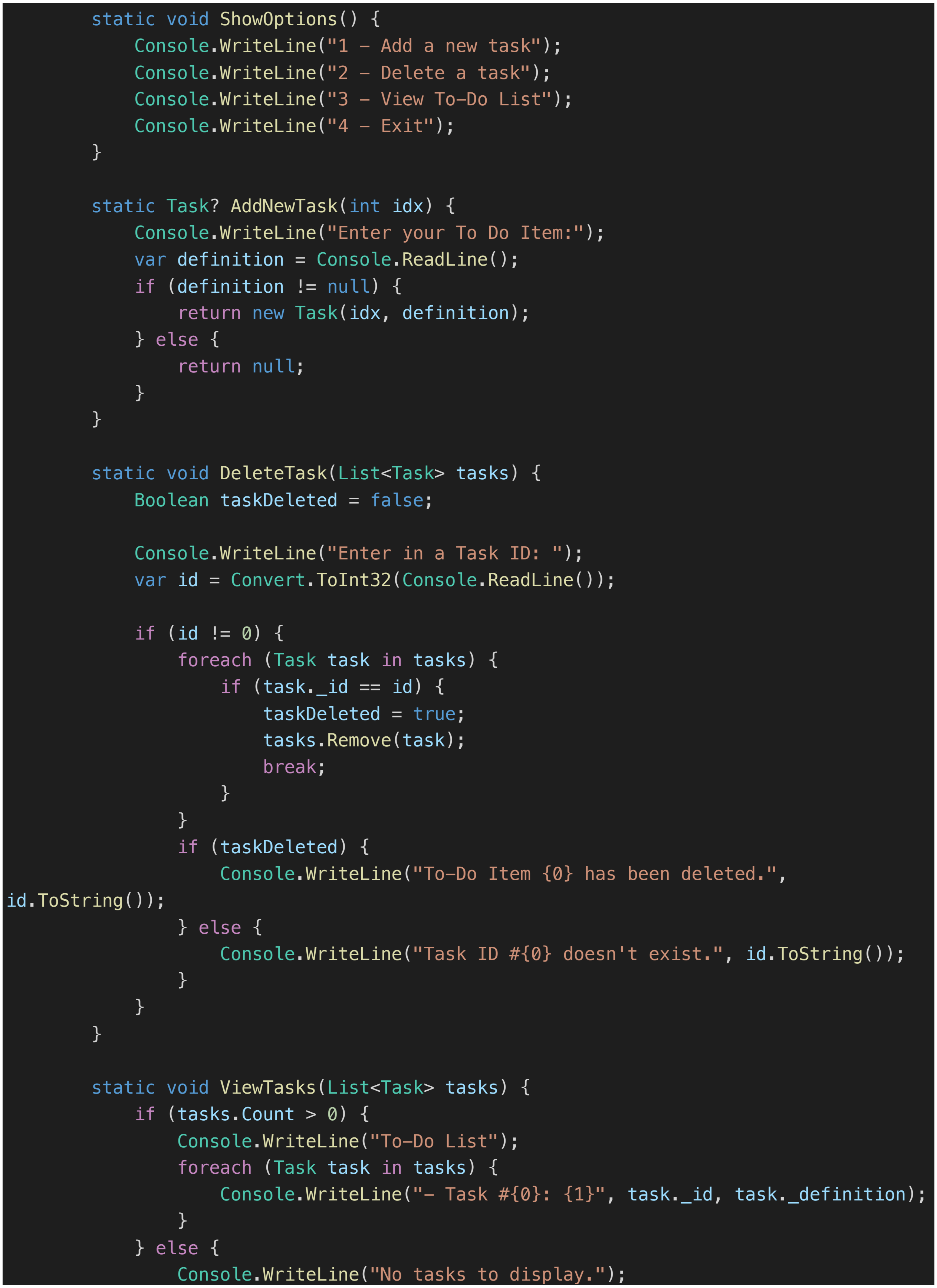
Console Output

```
geo@Geovanies-MBP Module2 % dotnet run
Welcome to the To-Do List Maker
1 - Add a new task
2 - Delete a task
3 - View To-Do List
4 - Exit
Please select a command:
1
Enter your To Do Item:
Walk the dogs
1 - Add a new task
2 - Delete a task
3 - View To-Do List
4 - Exit
Please select a command:
1
Enter your To Do Item:
Do the dishes
1 - Add a new task
2 - Delete a task
3 - View To-Do List
4 - Exit
Please select a command:
3
To-Do List
- Task #1: Walk the dogs
- Task #2: Do the dishes
1 - Add a new task
2 - Delete a task
3 - View To-Do List
4 - Exit
Please select a command:
2
Enter in a Task ID:
2
To-Do Item 2 has been deleted.
1 - Add a new task
2 - Delete a task
3 - View To-Do List
```

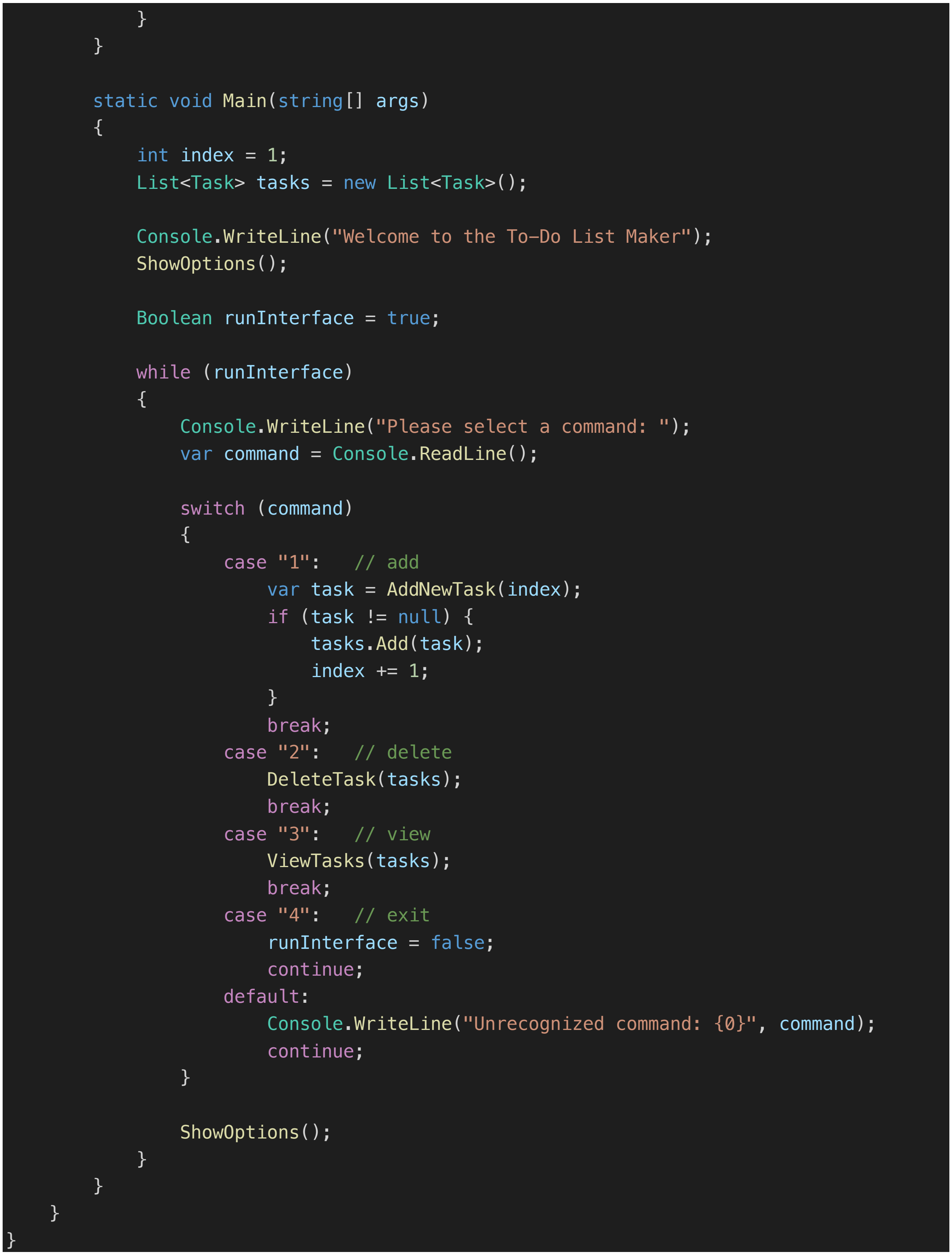
```
4 - Exit
Please select a command:
3
To-Do List
- Task #1: Walk the dogs
1 - Add a new task
2 - Delete a task
3 - View To-Do List
4 - Exit
Please select a command:
2
Enter in a Task ID:
7
Task ID #7 doesn't exist.
1 - Add a new task
2 - Delete a task
3 - View To-Do List
4 - Exit
Please select a command:
4
geo@Geovanies-MBP Module2 %
```

Source Code

```
using System;
using System.IO;
using System.Collections;
using System.Collections.ObjectModel;

namespace sd2.module2
{
    class Task
    {
        public int _id { get; set; }
        public string _definition { get; set; }

        public Task(int id, string definition)
        {
            _id = id;
            _definition = definition;
        }
    }

    class Program
    {
```



```
        static void ShowOptions() {
            Console.WriteLine("1 - Add a new task");
            Console.WriteLine("2 - Delete a task");
            Console.WriteLine("3 - View To-Do List");
            Console.WriteLine("4 - Exit");
        }

        static Task? AddNewTask(int idx) {
            Console.WriteLine("Enter your To Do Item:");
            var definition = Console.ReadLine();
            if (definition != null) {
                return new Task(idx, definition);
            } else {
                return null;
            }
        }

        static void DeleteTask(List<Task> tasks) {
            Boolean taskDeleted = false;

            Console.WriteLine("Enter in a Task ID: ");
            var id = Convert.ToInt32(Console.ReadLine());

            if (id != 0) {
                foreach (Task task in tasks) {
                    if (task._id == id) {
                        taskDeleted = true;
                        tasks.Remove(task);
                        break;
                    }
                }
                if (taskDeleted) {
                    Console.WriteLine("To-Do Item {0} has been deleted.",
id.ToString());
                } else {
                    Console.WriteLine("Task ID #{0} doesn't exist.", id.ToString());
                }
            }
        }

        static void ViewTasks(List<Task> tasks) {
            if (tasks.Count > 0) {
                Console.WriteLine("To-Do List");
                foreach (Task task in tasks) {
                    Console.WriteLine("- Task #{0}: {1}", task._id, task._definition);
                }
            } else {
                Console.WriteLine("No tasks to display.");
```



```
            }
        }

        static void Main(string[] args)
        {
            int index = 1;
            List<Task> tasks = new List<Task>();

            Console.WriteLine("Welcome to the To-Do List Maker");
            ShowOptions();

            Boolean runInterface = true;

            while (runInterface)
            {
                Console.WriteLine("Please select a command: ");
                var command = Console.ReadLine();

                switch (command)
                {
                    case "1":   // add
                        var task = AddNewTask(index);
                        if (task != null) {
                            tasks.Add(task);
                            index += 1;
                        }
                        break;
                    case "2":   // delete
                        DeleteTask(tasks);
                        break;
                    case "3":   // view
                        ViewTasks(tasks);
                        break;
                    case "4":   // exit
                        runInterface = false;
                        continue;
                    default:
                        Console.WriteLine("Unrecognized command: {0}", command);
                        continue;
                }

                ShowOptions();
            }
        }
    }
}
```