Program class: MAIN:  
namespace Car\_Tracker

{

internal class Program

{

static void Main(string[] args)

{

int option = 0;

Car car = new Car();

EventHandlers eventHandlers = new EventHandlers();

car.Event+=eventHandlers.handleNotificationEvent;

//Since car inherits tasks and tasks inherit Events class,

//we can directly use car object to access Events class parameter or methods.

while (option!=6)

{

try

{

Console.WriteLine("Choose one of the following:(\n1.Add Car\n2.View by Year\n3.Remove Car\n4.View All\n5.Update Status\n6.EXIT");

option = int.Parse(Console.ReadLine());

}

catch (Exception e) { Console.WriteLine("Please eneter a valid number between 1 to 6 :)"); }

switch (option) {

case 1:

int n = 1;

//adding car details to a list

while (n !=0)

{

n=car.insert();

}

break;

case 2:

//get car details by Year

try

{

Console.WriteLine("Which year car details you want?: ");

int y = int.Parse(Console.ReadLine());

car.GetCarsByYear(y);

}

catch (Exception e) { Console.WriteLine("Please eneter a valid Year :)"); }

break;

case 3:

//Delete car

Console.WriteLine("Enter name of the car you want to remove: ");

String cname = Console.ReadLine();

car.removeCar(cname);

break;

case 4:

//Display car details

car.details();

break;

case 5:

//Update Status

try

{

Console.WriteLine("Choose the following to update the status of the car: \n1)Sold\n2)Reserved\n3)Available");

int sn = int.Parse(Console.ReadLine());

if (sn <=0 || sn>3) { throw new Exception(); }

Console.WriteLine("Enter name of the car you want to update the status for:");

String carName = Console.ReadLine();

car.updateStatus(sn, carName);

}

catch (Exception e) { Console.WriteLine("Please choose a valid option between 1 to 3 :)"); }

break;

case 6:

car.saveAndExit();

return;

default:

Console.WriteLine(" Entered Option is invalid");

break;

}

}

}

}

}

CAR CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Car\_Tracker

{

public enum status

{

Sold ,

Reserved,

Available

}

internal class Car : Tasks

{

private string name;

private int year;

private string company;

private string model\_number;

private status cstatus;

public Car() { }

public Car(string Name, int Year, string Company, string Model\_number)

{

name=Name;

year=Year;

company=Company;

model\_number=Model\_number;

cstatus=status.Available;

}

public status Cstatus

{

set { cstatus = value; }

get { return cstatus; }

}

public string Model\_number

{

set { model\_number=value; }

get { return model\_number; }

}

public string Name

{

set { name=value; }

get { return name; }

}

public int Year

{

set { year=value; }

get { return year; }

}

public string Company

{

set { company=value; }

get { return company; }

}

public void display()

{

Console.WriteLine($"Car Name:{name}\nManufacturing Year: {year}\nCompany: {company}\nModel Number: {model\_number}\nStatus: {cstatus}:)------\n");

}

}

}

TASKS CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

using System.Text.Json;

using System.IO;

namespace Car\_Tracker

{

internal class Tasks:Events

{

public List<Car> cars = new List<Car>();

public int insert() {

Console.WriteLine("Enter Car Name: ");

string name = Console.ReadLine();

Console.WriteLine("Enter manufacturing Year: ");

int year = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Company: ");

string company = Console.ReadLine();

Console.WriteLine("Enter Model Number: ");

string model\_number = Console.ReadLine();

cars.Add(new Car(name, year, company, model\_number));

string msg = $"{name} added to the list successfully"; //Use of Event

startEvent(msg);

Console.WriteLine("Enter 1 to continue or Enter zero to stop adding car details: ");

int n=int.Parse(Console.ReadLine());

return n;

}

public void GetCarsByYear(int year) {

int count = 0;

foreach (Car c in cars)

{

if (c.Year == year) { count++;

c.display(); }

}

if (count == 0) {

string msg = $"Zero cars present for the entered year:{year}"; //Use of Event

startEvent(msg);

}

}

public void removeCar(string name) {

int count = 0;

for (int i = cars.Count - 1; i >= 0; i--) //cannot use foreach loop to remove. //

//can also use => cars.RemoveAll(car => car.Name == cname);

{

if (cars[i].Name == name)

{

count++;

cars.RemoveAt(i);

}

}

if (count!=0)

{

string msg = $"{name} successfully Deleted";

startEvent(msg);

}

else {

string msg = $"No Car found with the given name: {name}";

startEvent(msg);

}

}

public void details() {

Console.WriteLine("Car Details: < ------------->");

foreach (Car c2 in cars) { c2.display(); }

}

public void updateStatus(int n, String carName) {

int count = 0;

bool updated = false;

string msg = "";

for (int i = cars.Count - 1; i >= 0; i--)

{

if (cars[i].Name.Equals(carName, StringComparison.OrdinalIgnoreCase))

{

count++;

switch (n)

{

case 1:

if (cars[i].Cstatus != status.Sold)

{

cars[i].Cstatus = status.Sold;

updated = true;

}

else

{

msg = "Car is already Sold";

}

break;

case 2:

if (cars[i].Cstatus != status.Reserved)

{

cars[i].Cstatus = status.Reserved;

updated = true;

}

else

{

msg = "Car is already Reserved";

}

break;

case 3:

if (cars[i].Cstatus != status.Available)

{

cars[i].Cstatus = status.Available;

updated = true;

}

else

{

msg = "Car is already in Available status";

}

break;

default:

msg = "Invalid status code";

break;

}

}

}

if (count == 0)

{

msg = "No car found with the given name to update the status";

}

else if (updated)

{

msg = $"{carName} status updated successfully";

}

startEvent(msg);

}

public void saveAndExit() {

string path = "C:\\Users\\GBO3KOR\\Desktop\\C# edited files\\car\_tracker.json";

string json = JsonSerializer.Serialize(cars, new JsonSerializerOptions { WriteIndented = true });

File.WriteAllText(path, json);

string msg = "Data is saved. \nExiting the program.";

startEvent(msg);

}

}

}

EVENTS CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Car\_Tracker

{

internal class Events

{

//Delegates adn Events

public delegate void notify(string message);

public event notify Event;

public void startEvent(string message) {

Event?.Invoke(message);

}

}

}

EVENT HANDLER CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Car\_Tracker

{

internal class EventHandlers

{

public void handleNotificationEvent(string msg) {

Console.WriteLine(msg);

}

}

}