


To load information into the `LoadInfo` class from an XML file, you could use the `System.Xml` namespace in C#. Here's a simple way to read data from an XML file and load it into `LoadInfo`. This example assumes that you have a basic understanding of the XML file structure and know which elements to parse.

Here's an example of how you can do this:

## 1. Sample XML Structure

Assuming your XML file looks something like this:

xml


 Copy code

```
<CinemaData>
  <Cinema>
    <Name>Cinema 1</Name>
    <City>City 1</City>
    <Country>Country 1</Country>
  </Cinema>
  <Movie>
    <Name>Movie 1</Name>
    <ReleaseDate>2023-05-04</ReleaseDate>
    <AgeRating>PG-13</AgeRating>
  </Movie>
</CinemaData>
```

## 2. Modify the LoadInfo Class

Add properties to hold the data you want to load, and then create a method to read the XML file.

csharp

 Copy code

```
using System;
using System.Xml;

namespace CinemaBYT.Classes
{
    public class LoadInfo
    {
        public string CinemaName { get; set; }
        public string City { get; set; }
        public string Country { get; set; }
        public string MovieName { get; set; }
        public string ReleaseDate { get; set; }
        public string AgeRating { get; set; }

        // Method to Load data from XML
        public void LoadFromXml(string filePath)
        {
            XmlDocument doc = new XmlDocument();
            doc.Load(filePath);

            // Load Cinema information
            XmlNode cinemaNode = doc.SelectSingleNode("/CinemaData/Cinema");
            if (cinemaNode != null)
            {
                CinemaName = cinemaNode["Name"]?.InnerText;
                City = cinemaNode["City"]?.InnerText;
                Country = cinemaNode["Country"]?.InnerText;
            }
        }
    }
}
```

```
// Method to Load data from XML
public void LoadFromXml(string filePath)
{
    XmlDocument doc = new XmlDocument();
    doc.Load(filePath);

    // Load Cinema information
    XmlNode cinemaNode = doc.SelectSingleNode("/CinemaData/Cinema");
    if (cinemaNode != null)
    {
        CinemaName = cinemaNode["Name"]?.InnerText;
        City = cinemaNode["City"]?.InnerText;
        Country = cinemaNode["Country"]?.InnerText;
    }

    // Load Movie information
    XmlNode movieNode = doc.SelectSingleNode("/CinemaData/Movie");
    if (movieNode != null)
    {
        MovieName = movieNode["Name"]?.InnerText;
        ReleaseDate = movieNode["ReleaseDate"]?.InnerText;
        AgeRating = movieNode["AgeRating"]?.InnerText;
    }
}
}
```

[Copy code](#)

### 3. Usage Example

Once your `LoadInfo` class is set up, you can use it as follows:

```
csharp

class Program
{

```

[Copy code](#)

### 3. Usage Example

Once your `LoadInfo` class is set up, you can use it as follows:

```
csharp Copy code

class Program
{
    static void Main(string[] args)
    {
        LoadInfo loadInfo = new LoadInfo();

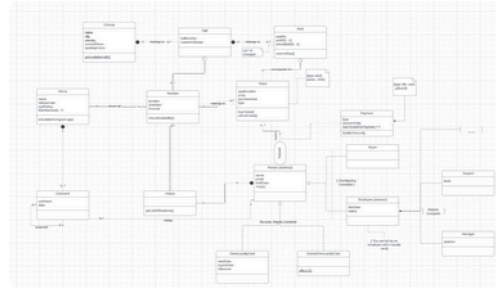
        // Replace with the path to your XML file
        string filePath = "path/to/your/file.xml";

        loadInfo.LoadFromXml(filePath);

        // Display loaded information
        Console.WriteLine($"Cinema Name: {loadInfo.CinemaName}");
        Console.WriteLine($"City: {loadInfo.City}");
        Console.WriteLine($"Country: {loadInfo.Country}");
        Console.WriteLine($"Movie Name: {loadInfo.MovieName}");
        Console.WriteLine($"Release Date: {loadInfo.ReleaseDate}");
        Console.WriteLine($"Age Rating: {loadInfo.AgeRating}");
    }
}
```

### Explanation

- **XmlDocument:** This is used to load and parse the XML file.
- **XPath:** `SelectSingleNode` is used with an XPath expression to locate specific nodes within the XML document.
- **Null Checks:** Ensure elements exist by using `?.InnerText` to avoid null reference exceptions.



Make every class in c#. Avoid methods.



Here is the C# code to represent all the classes shown in your diagram without the methods:

csharp

Copy code

```
public class Cinema
{
    public string Name { get; set; }
    public string City { get; set; }
    public string Country { get; set; }
    public string ContactPhone { get; set; }
    public string OpeningHours { get; set; }
    public List<Hall> Halls { get; set; }
}

public class Hall
{
    public int HallNumber { get; set; }
    public int NumberOfSeats { get; set; }
    public List<Seat> Seats { get; set; }
    public Cinema Cinema { get; set; } // Belongs to a Cinema
}
```



```
public class Seat
{
    public int SeatNo { get; set; }
    public bool IsVIP { get; set; }
    public bool IsAvailable { get; set; }
}

public class Movie
{
    public string Name { get; set; }
    public DateTime ReleaseDate { get; set; }
    public int AgeRating { get; set; }
    public List<string> ListOfGenres { get; set; }
}

public class Session
{
    public TimeSpan Duration { get; set; }
    public DateTime TimeStart { get; set; }
    public decimal Income { get; set; }
    public Movie Movie { get; set; }
    public Hall Hall { get; set; }
}
```

```
}


public class Ticket
{
    public int SeatNumber { get; set; }
    public decimal Price { get; set; }
    public DateTime PurchaseDate { get; set; }
    public TicketType Type { get; set; }
    public Session Session { get; set; }
}

public enum TicketType
{
    Adult,
    Senior,
    Child
}

public class Payment
{
    public PaymentType Type { get; set; }
    public DateTime PaymentDate { get; set; }
    public int MaxTicketPerPayment { get; set; } = 5;
}

public enum PaymentType
{
    Blik,
    Card,
    GiftCard
}
```



 Copy code

```
public abstract class Person
{
    public string Name { get; set; }
    public string Email { get; set; }
    public DateTime BirthDate { get; set; }
    public string PESEL { get; set; }
}

public class Buyer : Person
{
}

public abstract class Employee : Person
{
    public DateTime HireDate { get; set; }
    public decimal Salary { get; set; }
}

public class Manager : Employee
{
    public string Position { get; set; }
}

public class Support : Employee
{
    public string Level { get; set; }
}

public class Comment
{
    public string CommentText { get; set; }
    public DateTime Date { get; set; }
    public Movie Movie { get; set; }
}
```



```
public class OwnsLoyaltyCard
{
    public DateTime StartDate { get; set; }
    public DateTime ExpireDate { get; set; }
    public decimal Discount { get; set; }
}

public class DoesntOwnLoyaltyCard
{
    public int OfferLCO { get; set; }
}
```

### Key Notes:

- The diagram includes relationships, inheritance, and associations.
- Abstract classes (like `Person` and `Employee`) are defined accordingly.
- Enums were added for types like `TicketType` and `PaymentType`.
- I've avoided methods as per your request.

🔊 📄 👍 🗨️ ↺