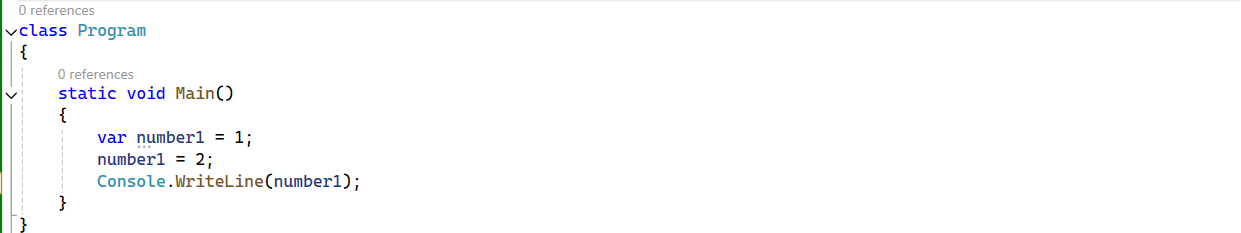
**HOW TO LEARN C# PROGRAMING LANGUAGE**

1. **C# Variable**
   1. **Var (global)**

**Var:** is the variable global, it can change the values of the variable, but it doesn’t security because it store variable on local.

**Note:** Var is can set a value in variable to other values



* 1. **Const**

**Const:** is used when you have a value that is constant and does not change throughout the program's execution.(ត្រូវ​បាន​ប្រើ​នៅ​ពេល​ដែល​អ្នក​មាន​តម្លៃ​ថេរ ហើយ​មិន​ផ្លាស់​ប្តូរ​ពេញ​មួយ​ការ​ប្រតិបត្តិ​របស់​កម្មវិធី។)

**Note:** Const is not set a value in variable to other values



1. **Data Type**
   1. **Boolean:** Represents true or false



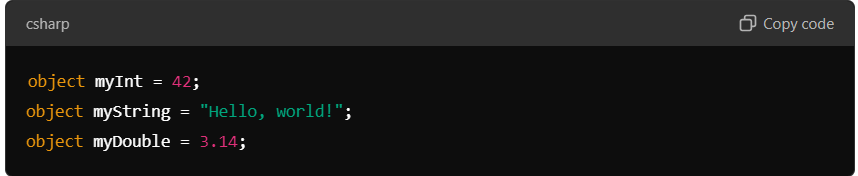
* 1. **Character:** Represents a single 16-bit Unicode character (0 to 65,535)

****

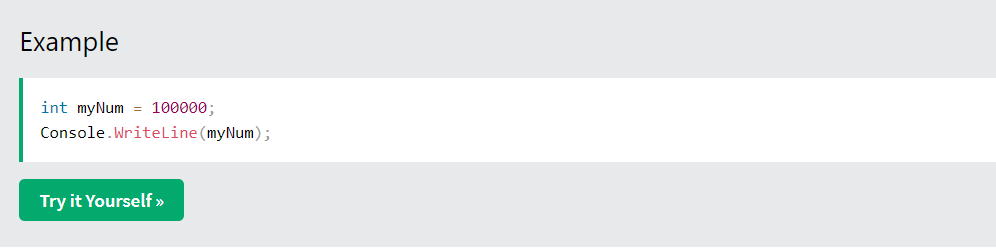
* 1. **String:** Represents a sequence of characters

****

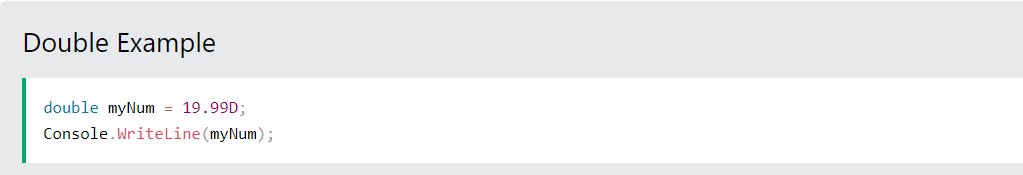
* 1. **Object:** The base type of all other types (both value and reference types)

****

* 1. **Number**

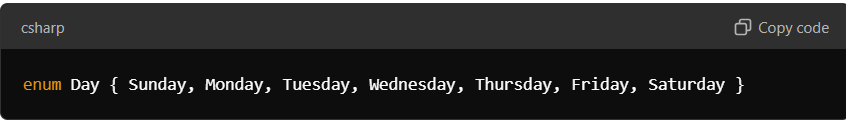
****

* 1. **Double**

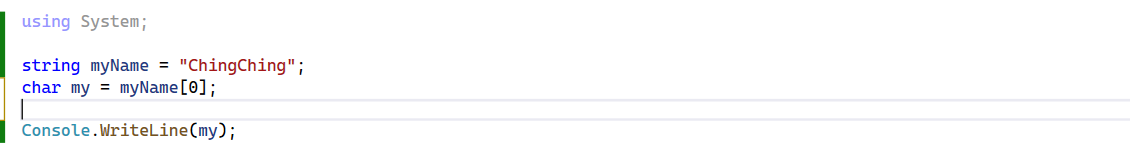
****

* 1. **Enum:** A special value type that defines a set of named constants

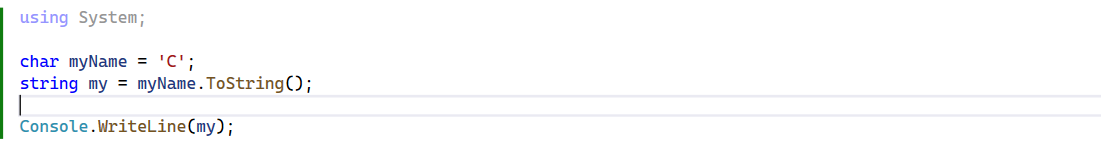
**Note: We used it for : set days, colors, months, direction**

****

1. **Casting Datatype**
   1. **Convert String to Char**

****

* 1. **Convert Char to String**

****

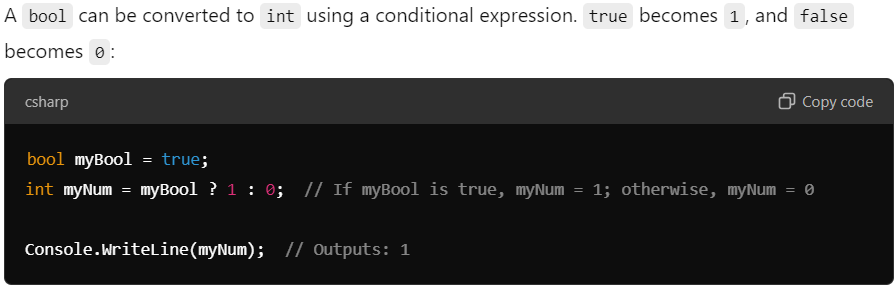
* 1. **Convert Double to Number**

****

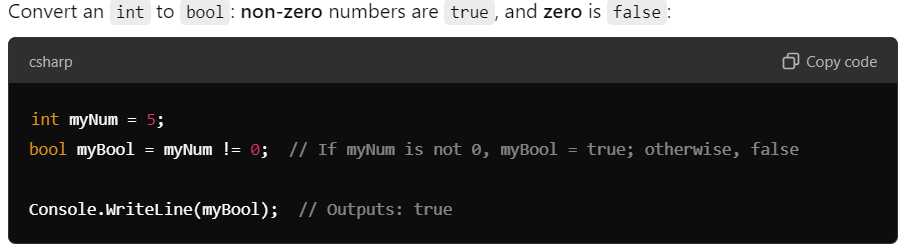
* 1. **Convert Number to Double**

****

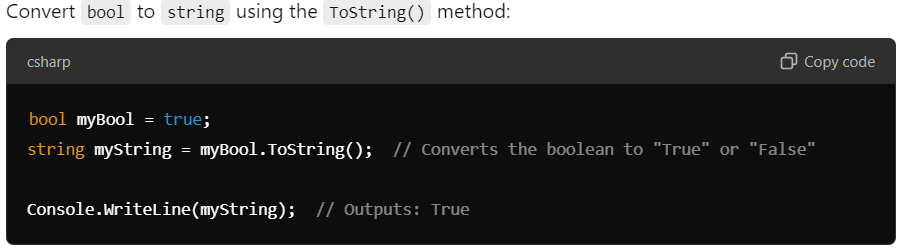
* 1. **Casting bool to int**

****

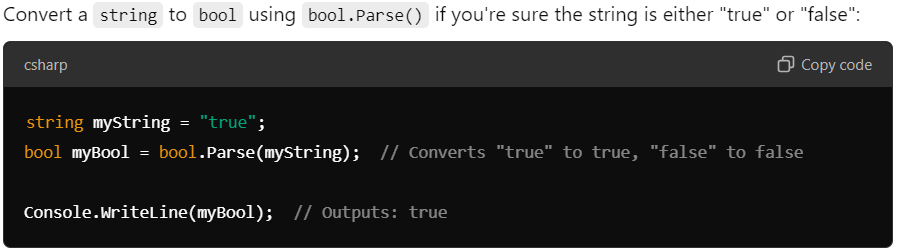
* 1. **Casting int to bool**

****

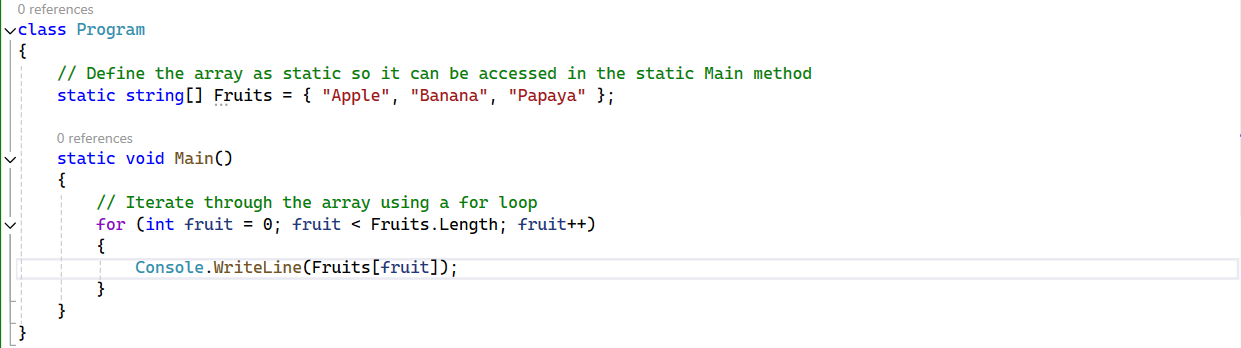
* 1. **Casting bool to string**

****

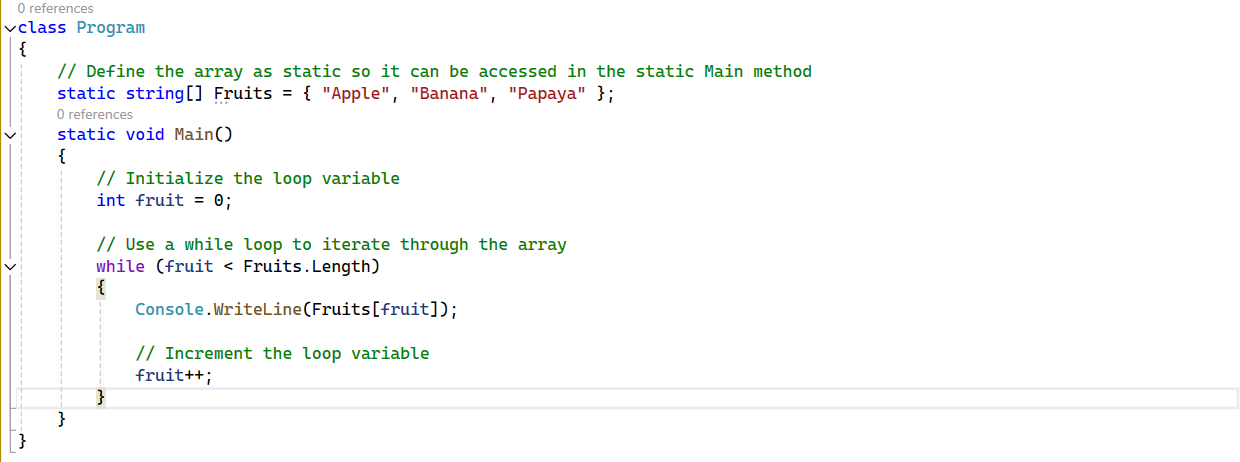
* 1. **Casting string to bool**

****

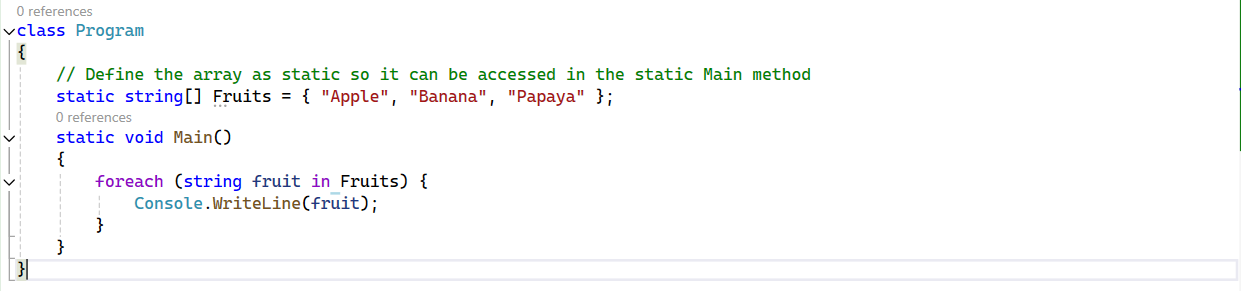
1. **Loop In C#**
   1. **For loop is get index.**

****

* 1. **While loop is get index too…**

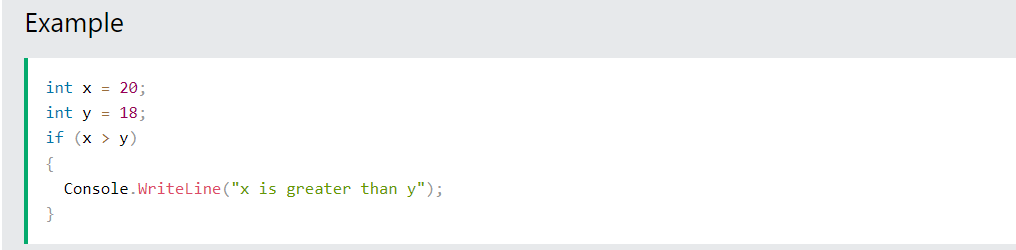
****

* 1. **Foreach loop is get values**

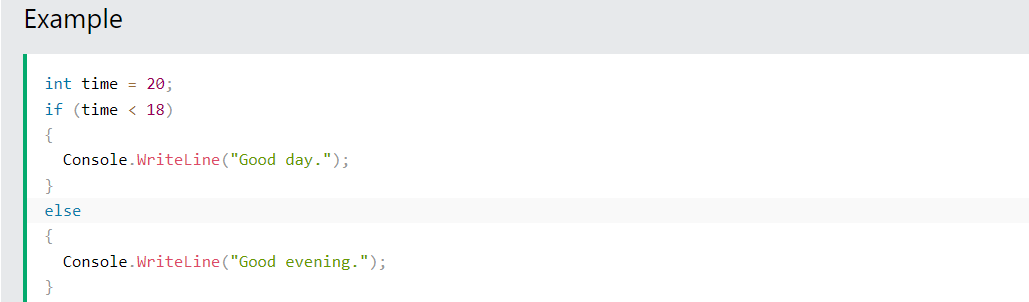
****

* 1. **recessive function**

1. **Condition**
   1. **Condition If**

****

* 1. **Condition Else**

****

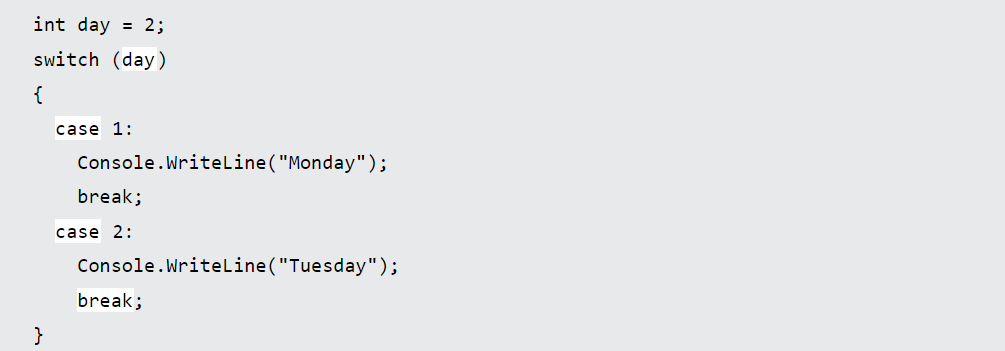
* 1. **Condition Else If**

****

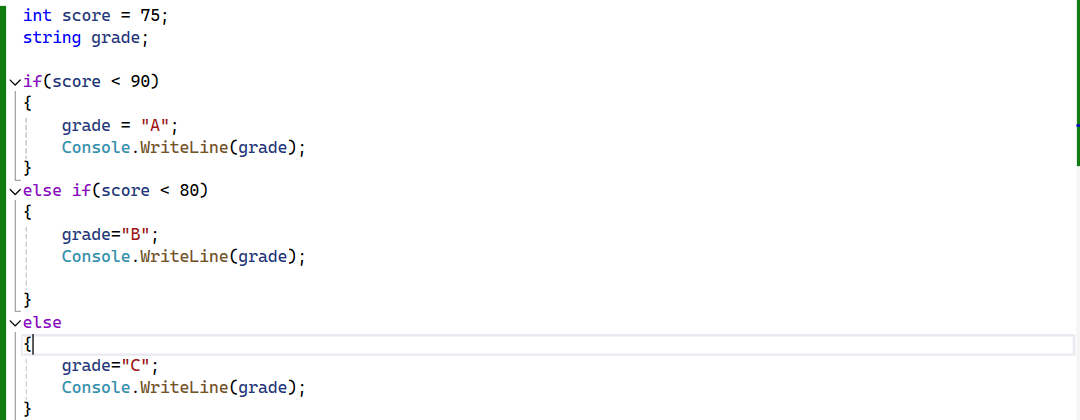
* 1. **Condition Switch**
     1. When do we use this condition?

when we need to evaluate a single expression or variable against multiple possible values and execute different code depending on which value is matched.

នៅពេលដែលយើងត្រូវវាយតម្លៃកន្សោមតែមួយ ឬអថេរធៀបនឹងតម្លៃដែលអាចធ្វើបានច្រើន ហើយប្រតិបត្តិកូដផ្សេងគ្នាអាស្រ័យលើតម្លៃដែលត្រូវគ្នា។



* 1. **Ternary Condition**

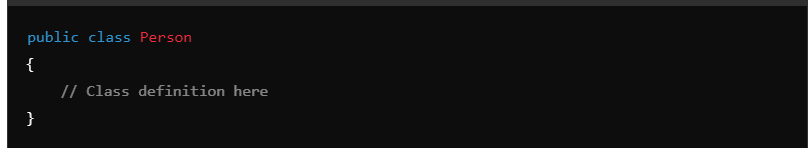


1. **Encapsulation**

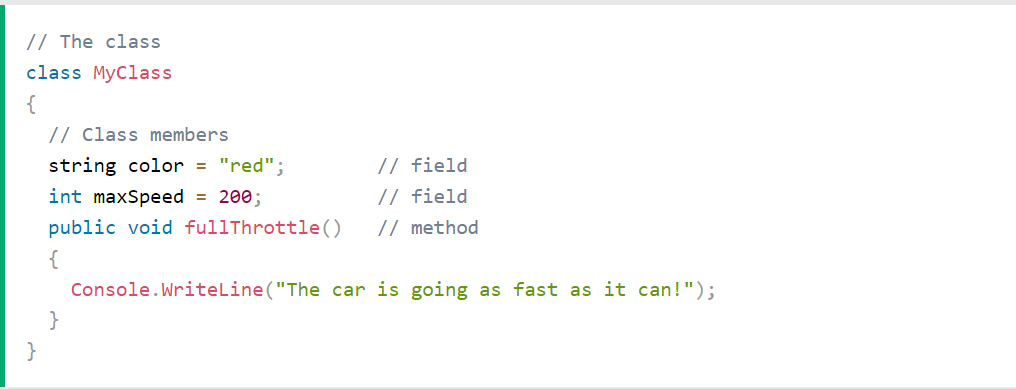
បណ្តុំនៃdata (attributes) និង method (function) ដែលដំណើរការលើ data ទៅជាឯកតាតែមួយ (class)។

Example: Class, member class, property, public, private, protected, interitance class, static class

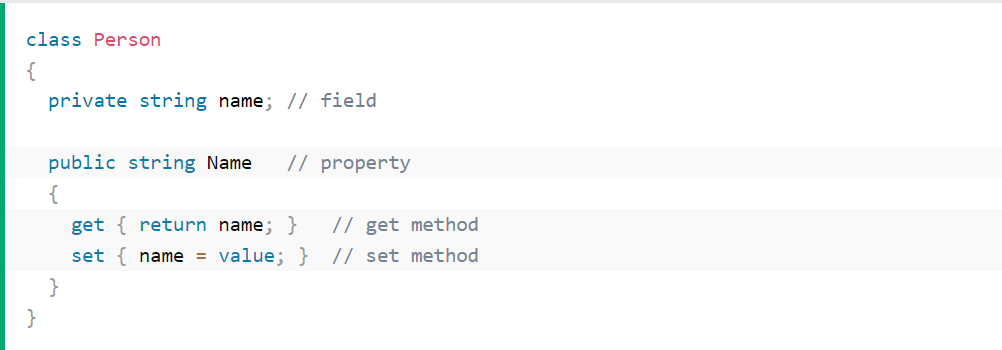
* 1. **Class**



* 1. **Members**: are variables defined in a class to hold data.



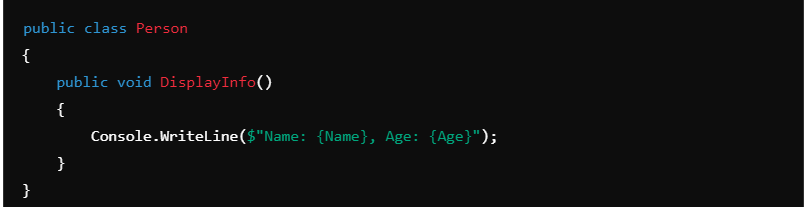
* 1. Property



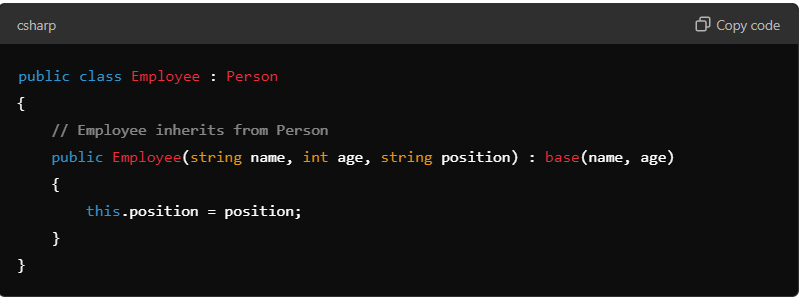
* 1. **Public**

**When to Use**: Use public when you want the member to be accessible from **anywhere**, including outside the class, in other classes, and in other assemblies.

**How it Works**: public members can be accessed by any class or method.



* 1. Inheritance



* 1. **Static**

**In C#, a static class is** a class that cannot be instantiated, meaning you can't create objects of this class. It is used to group methods, properties, or fields that are not tied to a specific instance of the class but rather to the class itself.

**When to Use a Static Class:**

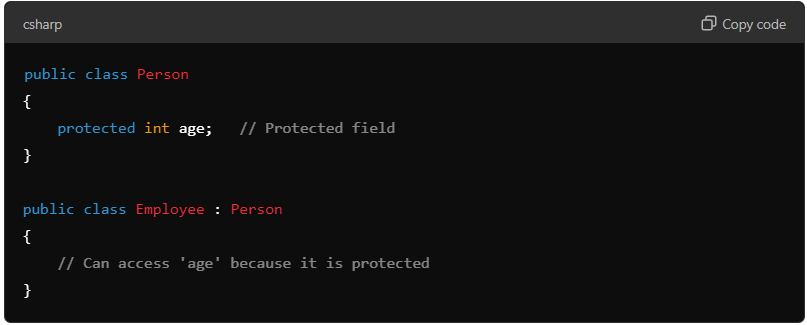
Utility or Helper Methods: When you need a set of functions that don’t depend on object data or state, such as mathematical calculations, string manipulations, or logging functionality.



* 1. Protected

**When to Use**: Use protected when you want the member to be accessible within the **same class** and **derived (inherited) classes**. Use only owner class and only inheritance from owner class

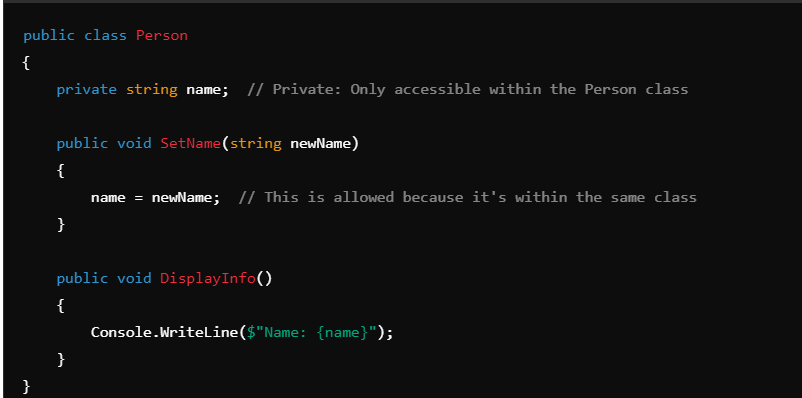
**How it Works**: protected members cannot be accessed outside of the class hierarchy, but derived classes can use them.



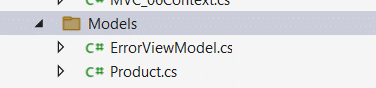
* 1. **Private**

**When to Use:** Use private when you want the member to be only accessible within the same class. Other classes cannot access private members directly.

**How it Works**: private members are hidden from outside access and can only be used within the class where they are defined.



1. **DB Connection**
   1. Right click on Models to created new Scaffolded Item



* 1. Used this command for connect database

Scaffold-DbContext "Server=DESKTOP-067JFQA\MSSQLSERVER1;Database=database\_name; user id = sa; pwd=123456789.;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Model -Context "your\_scaffolded item" -DataAnnotations

* 1. After you run the command on above, you will get the new Model

Note: in Models file must be have the file



* 1. Create the new controller

1. **ADODB(Data Set)**
2. **Fomating Code C# and Javascript**

Convention(Format coding)

1. Class in C# and in SQL(Table): You need to use capital Letter

example: class one word

- Class Person

example: class tow words

- Class PersonIfomation

2. Property(column): You need to use only small letter

example: class one word

- person

example: class tow words

- person\_ifomation

3. Variable: You need to use only small letter too...

example: class one word

- string person

example: class tow words

- string person\_ifomation

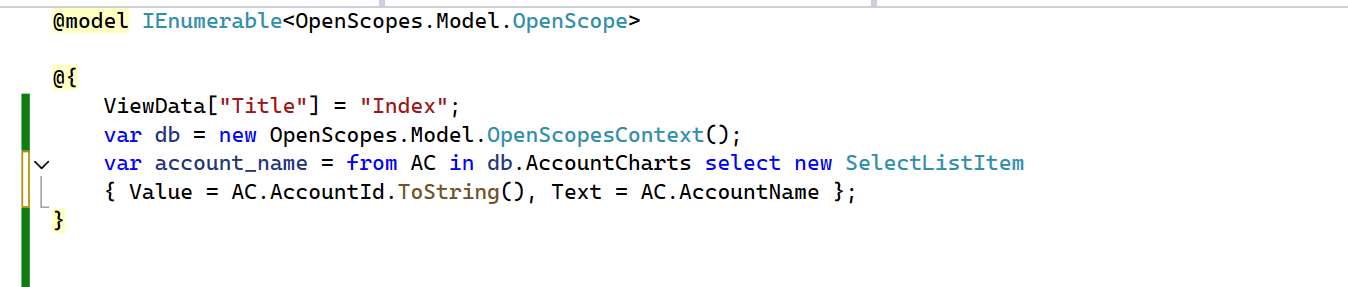
4. Function, Enum...: You need to use capital Letter(follow Class on above)

**Class and Property**

****

**How to bind and query data from SQL**

**In view code for select from SQL**

****

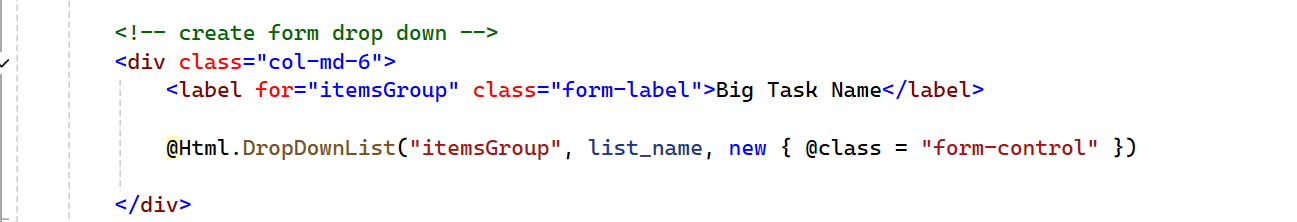
var db = new OpenScopes.Model.OpenScopesContext();

var account\_name = from AC in db.AccountCharts select new SelectListItem

{ Value = AC.AccountId.ToString(), Text = AC.AccountName };

**Code to display data**

@Html.DropDownList("itemsGroup", account\_name, new { @class = "form-control" })

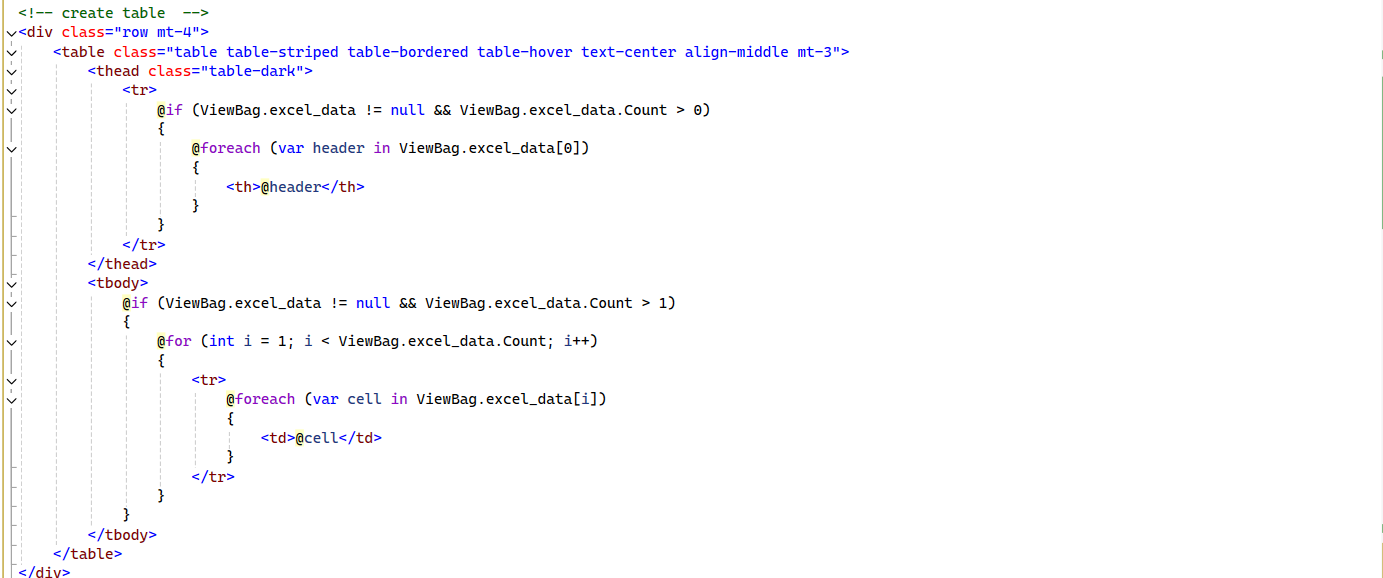


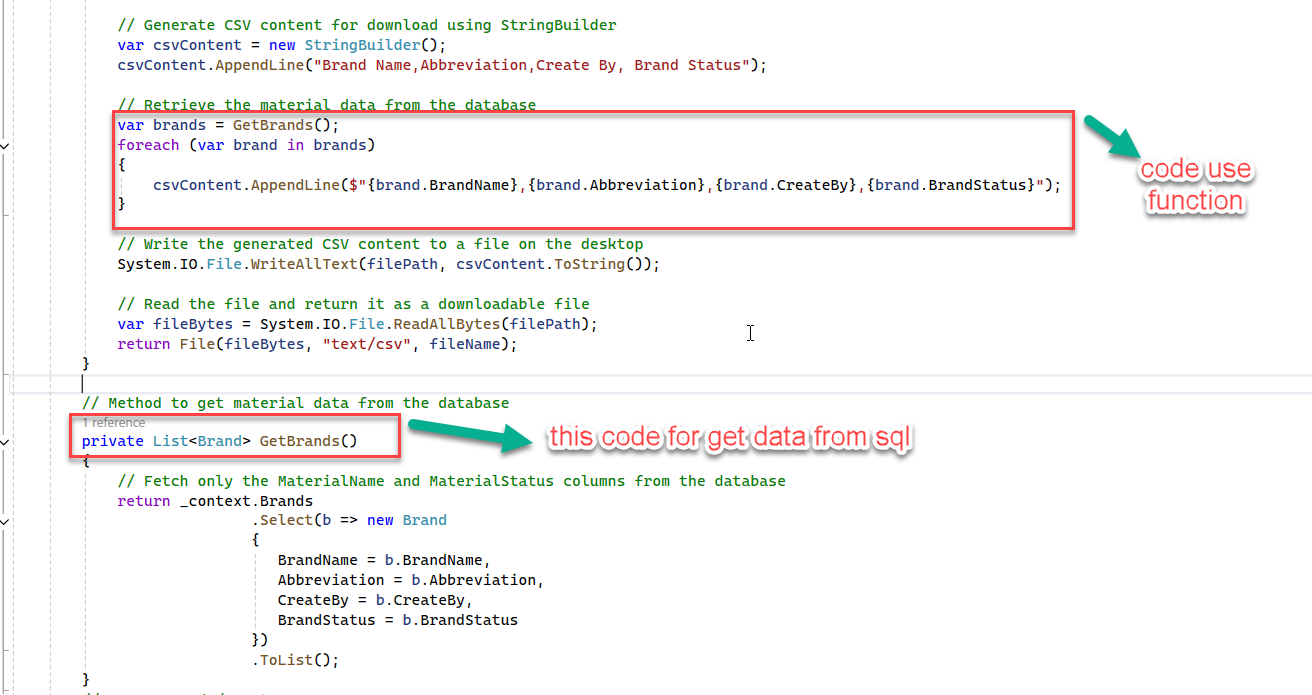
**How to Upload Excel file in MVC**

1. Install Excel Data Reader package
2. Here is code for Controller (copy code in Telegram)

****

1. Here is code for View page (copy in Telegram)

****



**// code for download excel file**

**public IActionResult Download()**

**{**

**// File name and path for saving to the desktop**

**var fileName = "Brand.csv";**

**var filePath = Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.Desktop), fileName);**

**// Generate CSV content for download using StringBuilder**

**var csvContent = new StringBuilder();**

**csvContent.AppendLine("Brand Name,Abbreviation,Create By, Brand Status");**

**// Retrieve the material data from the database**

**var brands = GetBrands();**

**foreach (var brand in brands)**

**{**

**csvContent.AppendLine($"{brand.BrandName},{brand.Abbreviation},{brand.CreateBy},{brand.BrandStatus}");**

**}**

**// Write the generated CSV content to a file on the desktop**

**System.IO.File.WriteAllText(filePath, csvContent.ToString());**

**// Read the file and return it as a downloadable file**

**var fileBytes = System.IO.File.ReadAllBytes(filePath);**

**return File(fileBytes, "text/csv", fileName);**

**}**

**// Method to get material data from the database**

**private List<Brand> GetBrands()**

**{**

**// Fetch only the MaterialName and MaterialStatus columns from the database**

**return \_context.Brands**

**.Select(b => new Brand**

**{**

**BrandName = b.BrandName,**

**Abbreviation = b.Abbreviation,**

**CreateBy = b.CreateBy,**

**BrandStatus = b.BrandStatus**

**})**

**.ToList();**

**}**