Garbage Collection

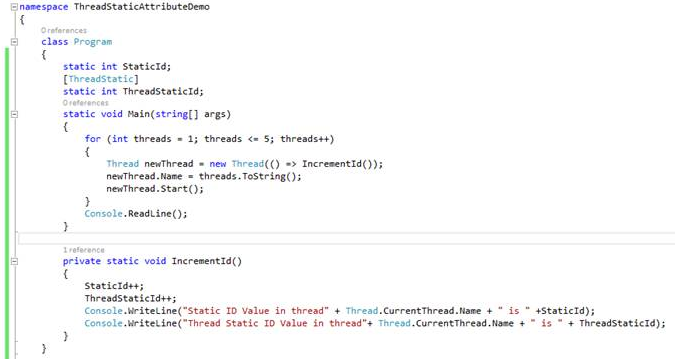
* Heap memory of the object only allow two types of variables: instance variable and instance read only variable.
* static and static readonly are out of bound for G.C and handled by CLR.
* \*\*G.C is only meant for instance and instance readonly variable.
* Myclass obj = new Myclass(), Here obj act as a refernce pointer in Stack which point to the memory addresso the class in HEAP.
* Four types of Heap Memory: Code Heap, SOH(Simple object Heap Memory), LOH(Large Object Heap), Process HEAP Memory.
* G.C is the feature of the CLR which clear the Memory allocated to the instance and instance readonly variables in the Heap Memory.
* CLR is also Known as **MScoriee.**
* Heap Memory is divided into Generations.
* If the size of the class memory is less than **85kb** then it is called as Simple Object HEAP Memory(SOH).
* If Class memory is equal to or larger than 85kb then it is called as large object Heap Memory.
* if Size is less than 85KB then it Goes to then Generation **0.**
* If greater than 85kb then it is stored in Generation 1.

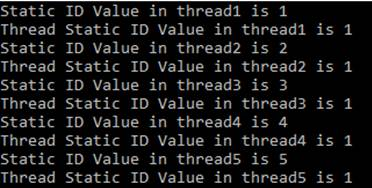
What for the Static and static readonly ?

* High Frequency Heap Memery (HFH)
* Static variables and static readonly variable is stored **in App domain(Cache memory for the Application)** by the CLR and HFH keep a reference of the variables in the App domain. **That's why they are said to be Global Variable.**
* A process can have multiple threads and a Single Instance can be allocated to all the threads belong the a Single process**.(Global Nature of static and static readonly).**
* **\*\*Cons:** Single instance and multiple threads is never safe and an attempt to change value by one thread can be reflected to other threads.

**Thread Static Architecture**

* Static variable can be made thread safe **One instance, One variable** is Followed.
* **eg.** Single PC multiple User, changes by one user is not reflected in other user Desktops.
* A static field marked with **ThreadStatic** Attribute is not shared between threads. A single instance per thread.
* eg. One department, One Canteen.
* Whenever we want to call multiple thread Simultaneously then use **Thread.Join().**This method blocks the Second thread till the execution of the first thread completed and it releases the





As we can see, static variable is shared among threads whereas ThreadStatic variable is unique to each thread and not shared.

Q1. What are Generations?

The heap is organized into generations so it can handle long-lived and short-lived objects. Garbage collection primarily occurs with the reclamation of short-lived objects that typically occupy only a small part of the heap. There are three generations of objects on the heap: •Generation 0. This is the youngest generation and contains short-lived objects. An example of a short-lived object is a temporary variable. Garbage collection occurs most frequently in this generation. Newly allocated objects form a new generation of objects and are implicitly generation 0 collections, unless they are large objects, in which case they go on the large object heap in a generation 2 collection. Most objects are reclaimed for garbage collection in generation 0 and do not survive to the next generation. •Generation 1. This generation contains short-lived objects and serves as a buffer between short-lived objects and long-lived objects. •Generation 2. This generation contains long-lived objects. An example of a long-lived object is an object in a server application that contains static data that is live for the duration of the process.

Q2. Why 85kb is set are the limit for the Heap Categorization?

Q3. Life of App domain? Scope of App domain?

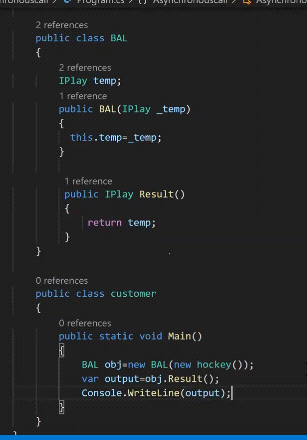
------------------------------------------------------------------------------------------------

***Interface***

* Interface is primararly made for finding the common Functionality Available in multiple classes of **Disimilar Type.**
* The Signature of the methods should not be changed.
* Eg. Entry time in an organisation should not be changed.
* **Dependecy Injection:** Technique in which an object or function get another object or function which it requires instead of creating them locally.
* We can pass Interface as a constructor paramters assigning to properties or passing as a method parameters to acheive Dependecy Injection.
* \*\*It can have **Properties, Events, Abstract Functions.**
* Abstract class find common functionality in **Similar Type** because of the Involvement of **Inheritance.**
* Astract class are meant for the same signature of the methods which has '**Is a'** Relationship between them.
* eg. ProjectManager and his team --> we use Abstract here, Employees of an Organisation of differnet teams --> We use Interface



* \*\* It gives us a High-end of Abstraction as We should not expose our function to the user and we are just declaring the method in Interface.
* We can pass Interface as a Constructor paramter.



* \*\* **It support a POLYMORPHIC Nature.**Same Function is defined By Multiple classes.
* \*\* Interface also Provide us with **Decoupling or Loose Coupling.**

------------------------------------------------------------------------------------------------

**Static Constructor**

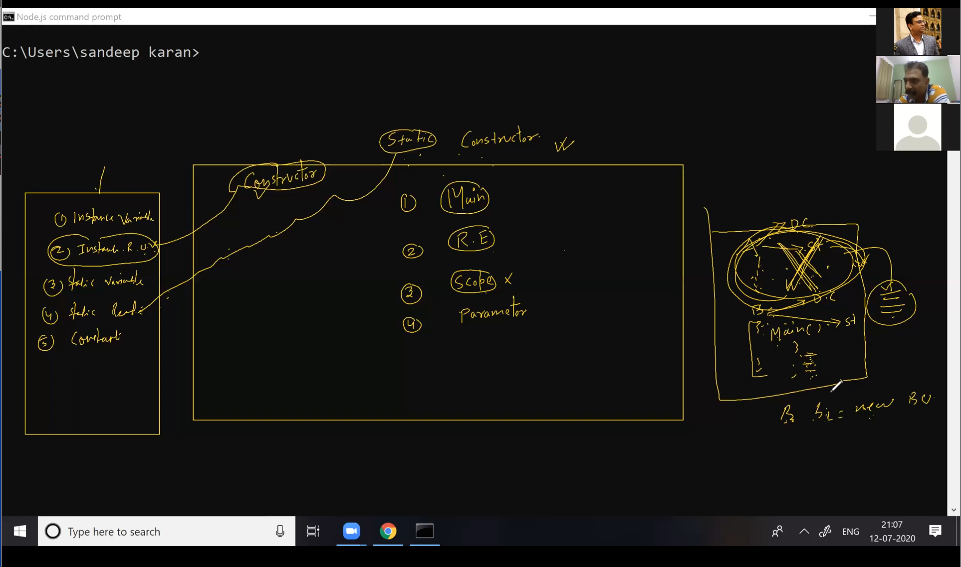
* Executes before Main so programmer can't have any control of the execution of the static constructor.
* if a constructor is defined as static then we can't define the scope of that consructor i.e we can't use public,private,internal.
* No parameterized constructor is formed of static constructor.we can't pass in any parameter in the static constructor.
* \*\* if we write new then constructor is called and if we marked the constructor as static then it is the responsibility of the CLR to call the static constructor.
* within an class the constructor can be called **n number of times. Everytime a new object is formed.**
* We can't have more than one static constructor and static constructor is called only one at the memory allocation of the class is done by the clr because there **is no Overloading of the static constructor.**
* **Static constructor is a one time effort.**
* **Major Usecase:**

1. to initialize the static readonly variable.
2. **We can't initialize the static readonly variable without static constructor.**

* constructor vs static-constructor

1. only constructor can initialize the instance readonly variable.
2. Only static constructor can intialize the static readonly constructor.

* Case study -1:



Suppose we are have two class A and B and Class B is inherited from class A and both of them has default and static constructor and class B has main function and it is empty.

if we run that code what is the output?

**Ans.** There is inheritance but if we are not making obj of the derieved class in main then no memory allocation is done for the parent class and no existence of the parent class.In this case only the static constructor of the derieved class is called.

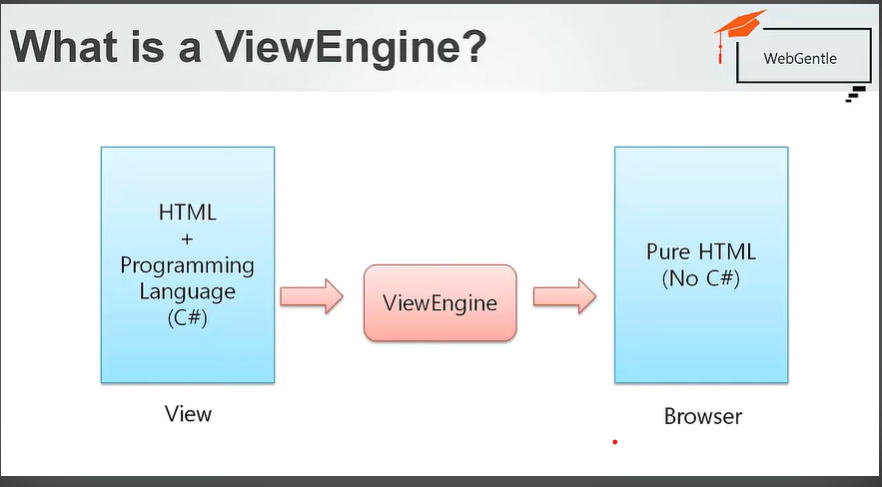
If the class A is marked as static then the CLR will automatically load the class otherwise unless we write new operator in class B then the class will not load and similarily cunstructor will not load.

* Instance read only vs Const

1. My addhar card is instance read only for me but Value of PI is a constant.
2. instance read only is constant for a particluar object only but Const is same for all irrespective of the object.

------------------------------------------------------------------------------------------------

**ViewEngine**

****

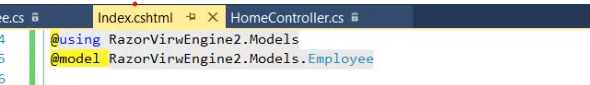
* View engine is repsonsible for creating HTML for views.
* It convert HTML+C# --> html for the browser.
* To find the corresponding view for action method(map action method to the corresponding view).
* used to write c# code on views.
* used for HTML helpers.
* used in strongly binding view.

**Razor**

* it is a view engine.
* view file extension for razor is .cshtml and .vbhtml.
* help in doing programming in html on view very easily.
* **Razor is a markup syntax for embedding server side code into webpages.**
* it begins with **@**
* if we want to use '@' in string then we want to add @@ instead of one.

**Directives in MVC**

* **@using -->** used to add model class and namespace.
* **@model -->** used to provide type of the model.



* we can use only one model on one view.
* MVC V1.0/2.0 uses ASPX as view engine after v4.0 we uses razor view engine.

**HTML Helpers**

* C# methods which are used to return HTML.
* using this we can render a text box, an area, image tag etc.
* MVC has Builtin HTML helpers.
* we can create custom helpers also.
* using HTML helpers a view can show model properties and can generate html as per the types of properties.
* Types of helpers:

1. Inline HTML Helpers

2. Built-in HTML Helpers

2.1 standard HTML helpers

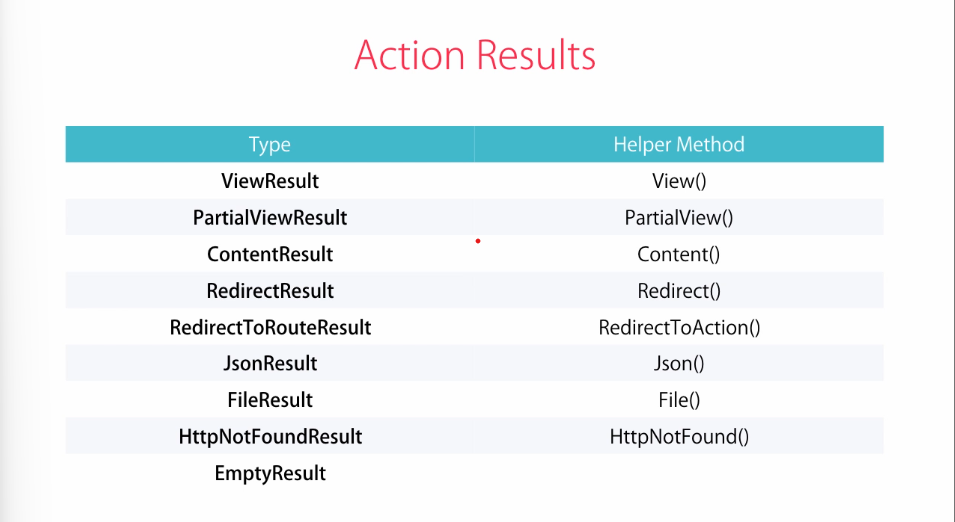
2.2 Strongly typed HTML helpers

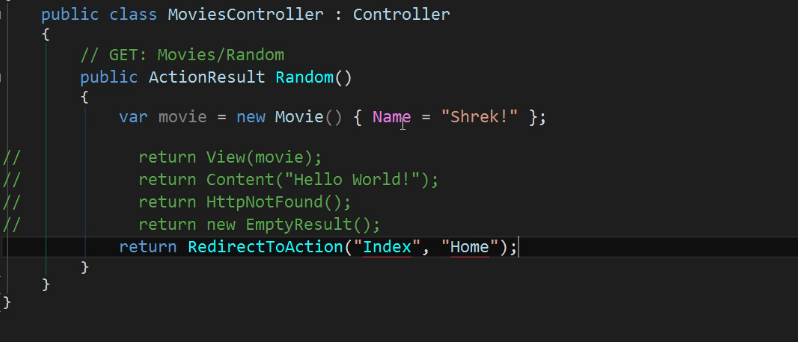
2.3 Templated HTML helpers

3. Custom HTML Helpers

--------------------------------------------------

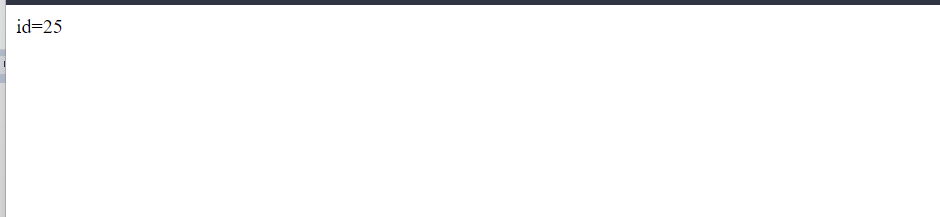
**Controller and views**

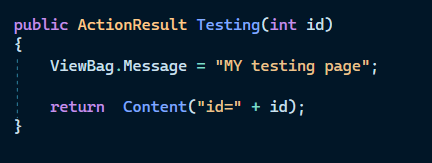




**Action Parameters**

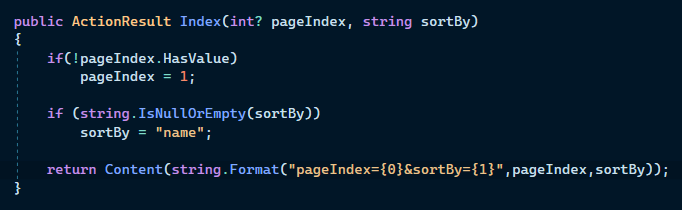
for: <https://localhost:44334/Home/Testing/25>





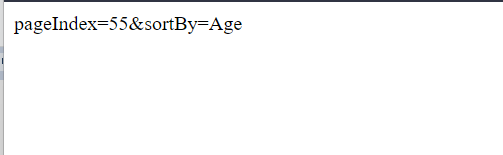
\*\*\*--------------\*\*\*

adding two action parameter



if no input provided then pageIndex =1 and sortPage='name'. we can also overide it.

url:<https://localhost:44334/Home/Index?pageIndex=55&sortBy=Age>



* **Conventional-based routing**

**routes.config.cs**

* routes.MapRoute(
* "MoviesByReleaseDate", // name of route(shoud be unique)
* "movies/released/{year}/{month}", // url of the route
* new {controller ="movies", action ="ByReleaseDate"}, // defining controller and action to be called
* // new { year = @"\d{4}", month = "**[Error! Hyperlink reference not valid.](\\\\d{2})**" } // new {year = "**[Error! Hyperlink reference not valid.](\\\\d{4})**", month="**[Error! Hyperlink reference not valid.](\\\\d{2)**}"} // applying constraints on length of year and month
* new { year = @"2015|2016", month = "**[Error! Hyperlink reference not valid.](\\\\d{2})**" } // now the year only take 2015 or 2016 as input
* );
* The above code is for the custom route where we applied constraints on year and month.

**Attribute-based routing(Recommended)**

* More advance form of routing introduced in MVC5.0.
* First we have to enable attribute routing by calling attribute route method before the defaut route method in **routeConfig.cs.**

routes.MapMvcAttributeRoutes(); // to define the attribute route

* now we need to write attribute based route on each actionMethod.

public class MoviesController : Controller

{

// GET: Movies

[Route("movies/released/{year}/{month:regex(**[Error! Hyperlink reference not valid.](\\\\d{2}):range(1,12)**)}")] // attribute route with parameters and constraints, month with length 4 and range of 1-12

public ActionResult ByReleaseDate(int year, int month)

{

return Content("Movie Year:" + year + " " + "Month is:" + month);

}

[Route("index")] //Simpe attribute route accessed by localhost:44311/index

public ActionResult Index()

{

return Content("I am index Action");

}

}

**Passing date to view**

* **using Viewbag**

**1.** It's just a wrapper on **ViewData** and it stores its data as dictionary type object(**Key-value pairs**) so we have to convert it to value type in View.

**MovieController.cs**

[Route("index")] //Simpe attribute route accessed by localhost:44311/index

public ActionResult Index()

{

var movieId = new Movie() { Id = 1 };

ViewBag.MovieId = movieId;

return View();

}

}

**MovieView.cs**

@{

ViewBag.Title = "Index";

var s = ViewBag.MovieId as WebApplication1.Models.Movie;

}

<h2>String value is: @s</h2>

* View data is generally not prefered due to its complex implimentation and boxing/unboxing of data.

------------------------------------------------------------------------------

------------------------------------------------------------------------------

**Authentication Filters**

* IAuthentication (Introduced in MVC5)
* IAuthorization
* IActionFilter
* IResultFilter
* IExceptionFilter