Constant vs read-only :

Both defines constant in application

Const key word

Eg: Const int a= 10

Readonly in b =20

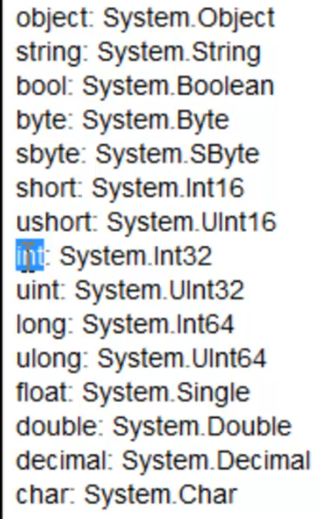
We cannot modify both values once defined and assigned value.

Const – compile time constant (absolute constant)

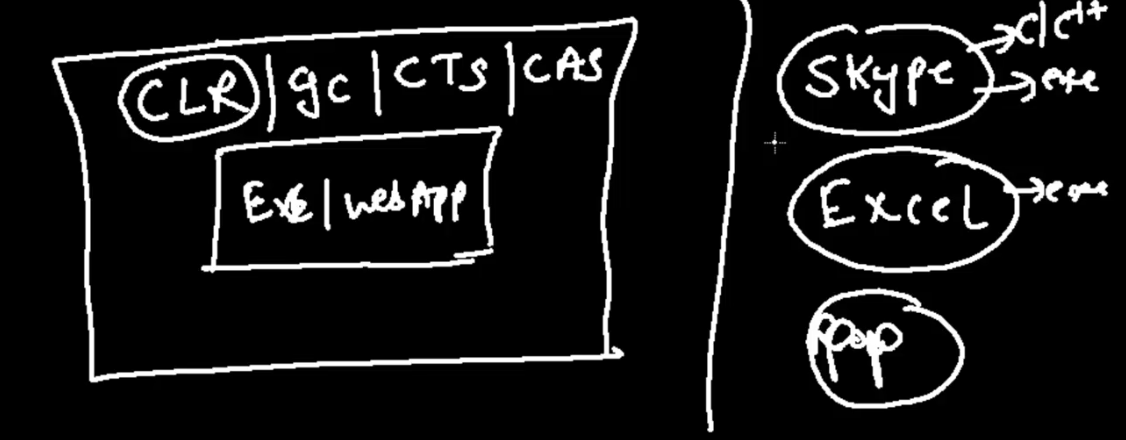
Readonly- run time constant (you need not assign at the time of declaration but can do assignment in constructor call)

String Vs string:

Both are same here small s is a allice for the caps s



Managed Vs Unmanaged code:



Code that runs under the control of the CLR is called managed code. Which has GC, cts, CAS runs only on >net frame work.

Application the runs on the run time of their development frame work other than CLR are called un manage code.

Can run unmanaged application from manage application but still it run under its own run time.

Debug Vs release mode:

1. Both are build modes
2. In debug mode there is no optimization of the code is done where in release optimization is done (Eg: exception with the stack trace result.)
3. In release mode all debug related code will be filtered from the source code.

IEnumerable Vs Ienumerator:

Both are same

I enumerable is synaptic sugar for the I enumerator

Internally it uses I enumerator

Difference is way we iterate through the list

List<int> ltr = new list<int >();

Ienumerable<int> ltr1=( Ienumerable<int>) ltr

Forech(var I in ltr1)

{

}

Ienumerator <int> ltr1=ltr.getEnumerator();

While(ltr1.moveNExt())

{

Ltr1.current.tostring();

}

1. In Ienumerable we directly iterate over list where as in Ienumerator we use GetEnumerator method to get enumerator instance > then use movenext to loop through items from the list

* Use current property to get the position of the current item from the list.

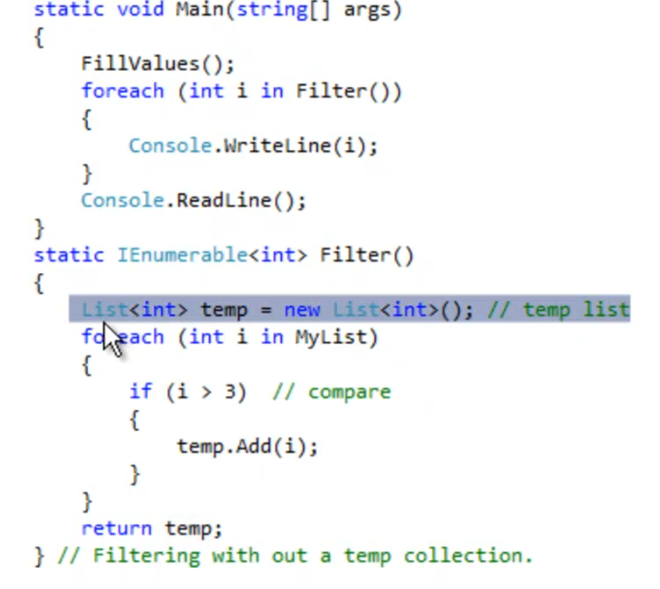
1. Ienumerator maintains the state i.e position of the cursor that its looping through.

Eg: display the list in range and pass it on to next method for processing

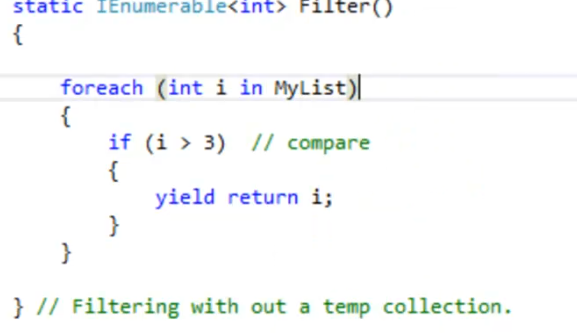
Yield in c#:

1. Used to custom iteration
2. Used to do Stateful iteration.

From the list filter only the numbers greater than 3

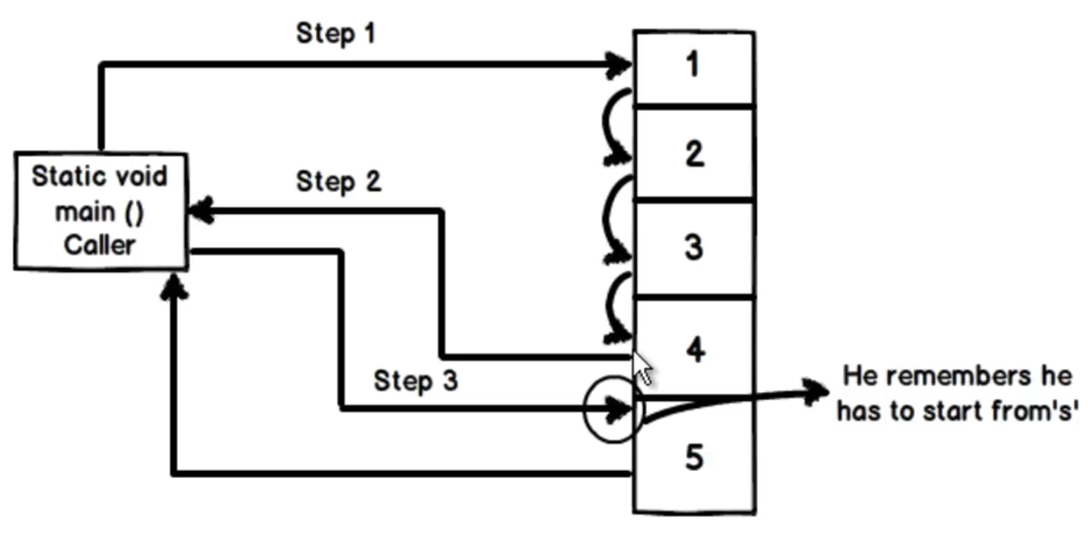


With the yield we can eliminate the temporary list to get the output



With yield its possible to return the yield of the operation from the successive methods calls and maintain the status and return the control back to the caller. With this there is flow of control from and to between caller and called method.

Method to call running total:



With yield code moves from caller to the called methods and called method to the caller to return result and resume from the point where it stopped.

