1. GC -- <https://www.codeproject.com/articles/1095402/garbage-collection-and-csharp>

**What is GC(Garbage Colloctor)?**

*Garbage collector*manages **allocation and reclaiming of memory**. GC (Garbage collector) makes a trip to the heap and collects all objects that are no longer used by the application and then makes them free from memory.

**Why we need it?**

When you create any object in C#, CLR (*common language runtime*) allocates memory for the object from heap. This process is repeated for each newly created object, but there is a limitation to everything, Memory is not un-limited and we need to clean some used space in order to make room for new objects

**How GC works?**

Machine generated alternative text:
GC works on managed heap, which is nothing but a block of memory to store objects, when garbage collection 
process is put in motion, it checks for dead objects and the objects which are no longer used, then it compacts 
the space of live object and tries to free more memory. 
Basically, heap is managed by different 'Generations', it stores and handles long-lived and short-lived objects, see 
the below generations of Heap: 
• O Generation (Zero): This generation holds short-lived objects, e.g., Temporary objects. GC initiates 
garbage collection process frequently in this generation. 
• 1 Generation (One): This generation is the buffer between short-lived and long-lived objects. 
• 2 Generation (Two): This generation holds long-lived objects like a static and global variable, that needs 
to be persisted for a certain amount of time. Objects which are not collected in generation Zero, are then 
moved to generation 1, such objects are known as survivors, similarly objects which are not collected in 
generation One, are then moved to generation 2 and from there onwards objects remain in the same 
generation. 

Machine generated alternative text:
When GC Gets Triggered? 
There are no specific timings for GC to get triggered, GC automatically starts operation on the following 
conditions: 
1. When virtual memory is running out of space. 
2. When allocated memory is suppressed acceptable threshold (when GC found if the survival rate (living 
objects) is high, then it increases the threshold allocation). 
3. When we call GC.C011ect() method explicitly, as GC runs continuously, we actually do not need to call 
this method. 

**What is managed and unmanaged resources or objects?**

Machine generated alternative text:
Managed objects are created, managed and under scope of CLR, pure .NET code managed by runtime, Anything 
that lies within .NET scope and under .NET framework classes such as string, int, bool variables are referred to 
as managed code. 
UnManaged objects are created outside the control of .NET libraries and are not managed by CLR, example of 
such unmanaged code is COM objects, file streams, connection objects, Interop objects. (Basically, third party 
libraries that are referred in .NET code.) 

**Clean up Unmanaged resources?**

When we create unmanaged objects, GC is unable to clear them and we need to *release such objects explicitly*when we finished using them.

Machine generated alternative text:
There are different ways to cleanup unmanaged resources: 
• Implement IDisposab1e interface and Dispose method 
'using' block is also used to clean unmanaged resources 
There are couple of ways to implement Dispose method: 
• Implement Dispose using 'SafeHand1e' Class (It is inbuilt abstract class which has 
'CriticalFina1izerObject' and 'IDisposab1e' interface has been implemented) 
Object . Finalize method to be override (This method is clean unmanaged resources used by particular 
object before it is destroyed) 

Machine generated alternative text:
Implement Dispose using overriding the 'Object. Finalize' method: 
Hide Shrink a 
class clsDispose Fin 
// Flag: Has Dispose already been caned? 
bool disposed 
false; 
// public implementation of Dispose pattern cal [able by consumers. 
public void Dispose() 
Disposel(true); 
GC . SuppressFina1ize(this); 
// protected implementation of Dispose pattern. 
protected virtual void Disposel(bool disposing) 
Copy Code 
if 
if 
(disposed) 
return; 
(disposing) 
// Free any other managed objects here. 
Free any unmanaged objects here. 
disposed = true; 
Disposel(false) ; 

Machine generated alternative text:
Re-view 
Garbage collector manages allocation and reclaim of memory. 
GC works on managed heap, which is nothing but a block of memory to store objects. 
• There is no specific timings for GC to get triggered, GC automatically start operation. 
Managed objects are created, managed and under scope of CLR. 
Unmanaged objects are wrapped around operating system resources like file streams, database 
connections, network related instances, handles to different classes, registries, pointers, etc. 
Unmanaged resources can be cleaned-up using 'Dispose' method and 'using' statement. 