**What is Stack(memory)?**

Memory is implemented using stack data structure which works on last in first out basis

1. Items inserted and removed from one end of the DS
2. Item inserted last will be the item to be removed
3. Insert operation called push, remove operation called pop.

**Stack stores various types of data:**

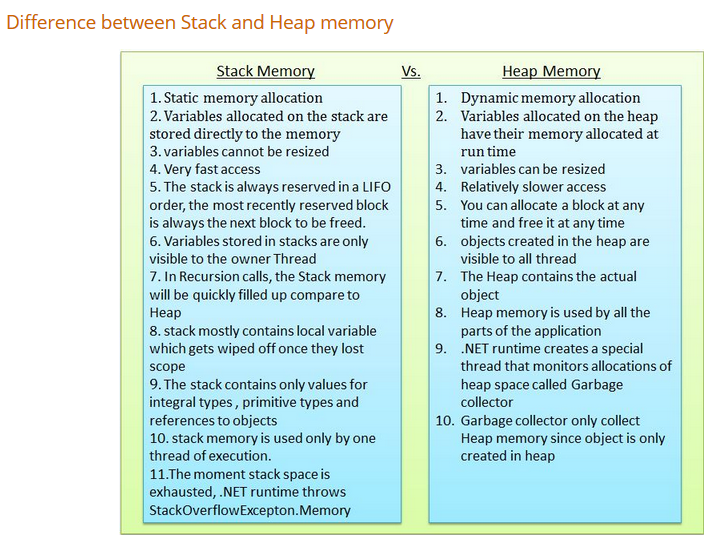
1. Values types (values of the certain types will be saved)
2. Function parameters
3. Programs current execution environment.

### What is Heap Memory?

Area in the memory where the chunks are allocated to store certain kind of data objects. Unlike stack objects can be inserted and removed from any part of the heap.

You can allocate chunks for the data object but you cannot delete them explicitly. CLRs garbage collector will perform the clean up when it comes to know that orphaned object are no more referred in you code.

Difference:





While running an application lot of objects created . It manipulates the object and at some point of time object is no longer needed such object becomes the candidate for the garbage collection.

CLR stores data in two memory while running the code

1.Stack

2.heap

Stack keeps track of what is currently executing in your code like local variables.

Heap keeps track of the objects

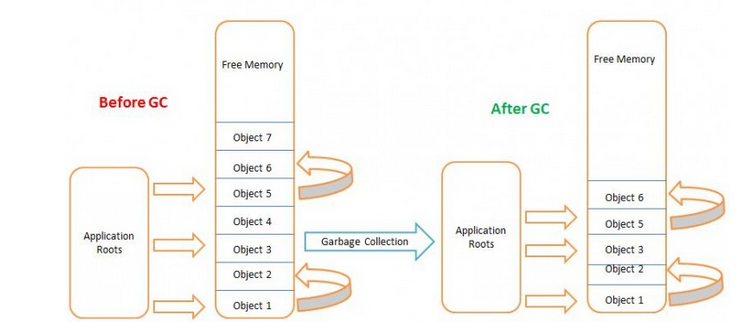
Every object on the heap there will be a reference on stack.

Garbage collector cleans up the heap when there is no enough room in the heap to create objects.

Heap is managed by the garbage collector.

In an unmanaged environment we need to keep track of all the objects allocated on heap and clean them up when it is candidate for garbage collection.

How garbage collection works



Before GC application root depends on obj1, obj13 obj5

obj1-> obj13

obj5-> obj6

**When running GC:**

1.Marking phase:

Mark all objects candidate for the GC

Got though the entire heap for all the reference object, parameter referring heap object, CPU register and other items that has reference to heap object.

2.Relocating phase:

For each object, GC marks will mark object to which the reference point as it is.

3.Compact phase:

GC will perform the compacting for the element in heap. Update the program reference

Then update the heap so that the program can allocation new objects from the heap

### After the Garbage collector runs:

Obejct7 and Obejct4 are released from memory because there are no references to it.

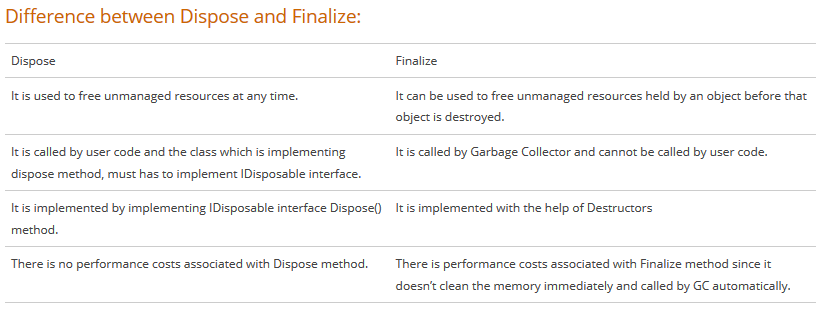
When object is destroyed all the allocated memory and resource help by the object will be released.

Using destructor and dispose method we can free the managed and unmanaged resource held by the object

Finalization increases then Object life. Because it has to run the finalization code. .net keep reference of the object in special finalization queue. An additional thread run all the finalizer at a time deemed appropriate based on the execution context.

This delays the GC of the type have finalizer.

Difference:



### Using Weak References:

### Scenario where object takes lot of memory. But wanted it to be recreated when needed. Cache scenario where data is kept in the memory but still you want it to be garbage collected

### 

### Way of having pointer to a