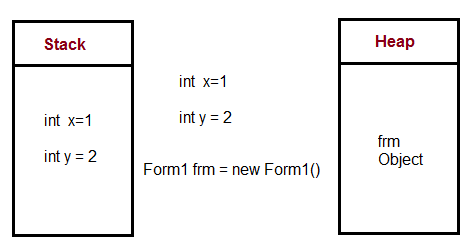
**Stack:** It is used for static memory allocation, stored in the computer's RAM

Variables allocated on the stack are stored directly to the memory and access to this memory is very fast, and its allocation is dealt with when the program is compiled.

**Stack is thread specific**

**When to use**

You can use the stack if you know exactly how much data you need to allocate before compile time and it is not too big.



**Heap:** for dynamic memory allocation, stored in the computer's RAM .

Variables allocated on the heap have their memory allocated at run time and accessing this memory is a bit slower, but the heap size is only limited by the size of virtual memory. Element of the heap have no dependencies with each other and can always be accessed randomly at any time. You can allocate a block at any time and free it at any time. This makes it much more complex to keep track of which parts of the heap are allocated or free at any given time.

**Heap is application specific.**

**When to use**

You can use heap if you don't know exactly how much data you will need at runtime or if you need to allocate a lot of data.

**Note:** In a multi-threaded situation each thread will have its own completely independent stack but they will share the heap. Stack is thread specific and Heap is application specific. The stack is important to consider in exception handling and thread executions.

**Garbage collection:**

Garbage collection is a process of releasing the memory used by the objects, when they are no longer referenced, garbage collector serves as an automatic memory manager. It provides the following benefits:

Enables you to develop your application without having to free memory.

Allocates objects on the managed heap efficiently.

Reclaims objects that are no longer being used, clears their memory, and keeps the memory available for future allocations. Managed objects automatically get clean content to start with, so their constructors do not have to initialize every data field.

Provides memory safety by making sure that an object cannot use the content of another object.