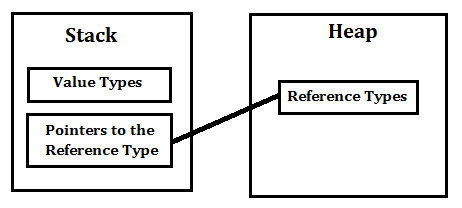
The Types in .NET Framework are either treated by Value Type or by Reference Type. A Value Type holds the data within its own memory allocation and a Reference Type contains a pointer to another memory location that holds the real data. Reference Type variables are stored in the heap while Value Type variables are stored in the stack.



**Value Type:**

A Value Type stores its contents in memory allocated on the stack. When you created a Value Type, a single space in memory is allocated to store the value and that variable directly holds a value. If you assign it to another variable, the value is copied directly and both variables work independently. Predefined datatypes, structures, enums are also value types, and work in the same way. Value types can be created at compile time and Stored in stack memory, because of this, Garbage collector can't access the stack, e.g. int x = 10. Here the value 10 is stored in an area of memory called the stack.

**Reference Type:**

Reference Types are used by a reference which holds a reference (address) to the object but not the object itself. Because reference types represent the address of the variable rather than the data itself, assigning a reference variable to another doesn't copy the data. Instead it creates a second copy of the reference, which refers to the same location of the heap as the original value. Reference Type variables are stored in a different area of memory called the heap. This means that when a reference type variable is no longer used, it can be marked for garbage collection. Examples of reference types are Classes, Objects, Arrays, Indexers, Interfaces etc., e.g. int[] iArray = new int[20];

In the above code the space required for the 20 integers that make up the array is allocated on the heap.

Stack is used for static memory allocation and Heap for dynamic memory allocation, both stored in the computer's RAM.