1. Classes are Reference types and structures are Values types.
2. Struct creates in stack, whereas class creates in the heap.
3. Classes are usually used for large amounts of data, whereas structures are usually used for smaller amounts of data.
4. Classes can be inherited whereas structures not.
5. A structure couldn't be null like a class.
6. A structure couldn't have a destructor such as a class.
7. A structure can't be abstract, a class can.
8. You cannot override any methods within a structure except the following belonging to the type object:  
   * Equals()
   * GetHashCode()
   * GetType()
   * ToString()

And the other polymorphism technique used for structures is implementing interfaces.

1. Declared events within a class are automatically locked and then they are thread safe, in contrast to the structure type where events can't be locked.
2. A structure must always have the default parameter less constructor defined as public but a class might have one, so you can't define a private parameter-less constructor as in the following:  
     
   struct Me  
       {  
           private Me()// compile-time error  
           {  
           }  
       }  
          
   class Me  
            {  
               private Me()// runs Ok{  
            }
3. A static constructor is triggered in the case of a class but not in the case of a structure as in the following:  
     
   struct myStructure  
       {  
           static myStructure()   
           {  
               Console.WriteLine("This is me a structure");  
           }  
       }  
       class myClass

    {  
        static myClass()  
        {  
            Console.WriteLine("This is me a class");  
        }  
    }

    class Program  
    {  
        static void Main(string[] args)  
        {  
           myStructure s =new myStructure();//Nothing happen  
           myClass c =new myClass();//Will out put This is me a class  
           Console.Read();  
        }  
    }

1. The structure can't contain a volatile field whereas the class can
2. You can't use size of with classes but you can with structures
3. Fields are automatically initialized with classes to 0/false/null whereas structures are not
4. Fields can't be directly instantiated within structures but classes allow such operations as in the following:  
     
   struct myStructure  
       {  
           publicstring x = 2;//Not allowed   
       }  
       class myClass  
       {  
           publicstring x = 2;//Allowed  
       }
5. Structures and classes don't adopt the same approach for the System.Object.Equals() method.