Dear Friends,

Hi I am satish marwat, this documents contains all the important questions that usually asked during the .NET interview, I had downloaded all the material from the Internet from various websites and collected to form a single film, u will find few repeated questions also, all the material are from the various websites, so I had just bind it into a single file.

So for any mistake I am not responsible, this is just for the view purpose. My view was only to collect a material to a single file.

Please, if u find any mistake in this file, please contact me to my email address satishcm@gmail.com, so that I can able to correct it. **ALL THE BEST**

Thanks

Satish ☺

.NET FRAME WORK

Introduction

1.1 What is .NET?

.NET is a general-purpose software development platform, similar to Java. At its core is a virtual machine that turns intermediate language (IL) into machine code. High-level language compilers for C#, VB.NET and C++ are provided to turn source code into IL. C# is a new programming language, very similar to Java. An extensive class library is included, featuring all the functionality one might expect from a contempory development platform - windows GUI development (Windows Form s), database access (ADO.NET), web development (ASP.NET), web services, XML etc.

1.2 When was .NET announced?

Bill Gates delivered a keynote at Forum 2000, held June 22, 2000, outlining the .NET 'vision'. The July 2000 PDC had a number of sessions on .NET technology, and delegates were given CDs containing a pre-release version of the .NET framework/SDK and Visual Studio.NET.

1.3 What versions of .NET are there?

The final version of the 1.0 SDK and runtime was made publicly available around 6pm PST on 15-Jan-2002. At the same time, the final version of Visual Studio.NET was made available to MSDN subscribers.
.NET 1.1 was released in April 2003 - it's mostly bug fixes for 1.0.
.NET 2.0 is expected in 2005.

1.4 What operating systems does the .NET Framework run on?

The runtime supports Windows Server 2003, Windows XP, Windows 2000, NT4 SP6a and Windows ME/98. Windows 95 is not supported. Some parts of the framework do not work on all platforms - for example, ASP.NET is only supported on XP and Windows 2000/2003. Windows 98/ME cannot be used for development.
IIS is not supported on Windows XP Home Edition, and so cannot be used to host ASP.NET. However, the ASP.NET Web Matrix web server does run on XP Home.
The .NET Compact Framework is a version of the .NET Framework for mobile devices, running Windows CE or Windows Mobile.
The Mono project has a version of the .NET Framework that runs on Linux.

1.5 What tools can I use to develop .NET applications?

There are a number of tools, described here in ascending order of cost:

- The .NET Framework SDK is free and includes command-line compilers for C++, C#, and VB.NET and various other utilities to aid development.
- ASP.NET Web Matrix is a free ASP.NET development environment from Microsoft. As well as a GUI development environment, the download includes a simple web server that can be used instead of IIS to host ASP.NET apps. This opens up ASP.NET development to users of Windows XP Home Edition, which cannot run IIS.
- Microsoft Visual C# .NET Standard 2003 is a cheap (around $100) version of Visual Studio limited to one language and also with limited wizard support. For example, there's no wizard support for class libraries or custom UI controls. Useful for beginners to learn with, or for savvy developers who can work around the deficiencies in the supplied wizards. As well as C#, there are VB.NET and C++ versions.
- Microsoft Visual Studio.NET Professional 2003. If you have a license for Visual Studio 6.0, you can get the upgrade. You can also upgrade from VS.NET 2002 for a token $30. Visual Studio.NET includes support for all the MS languages (C#, C++, VB.NET) and has extensive wizard support.

At the top end of the price spectrum are the Visual Studio.NET 2003 Enterprise and Enterprise Architect editions. These offer extra features such as Visual Sourcesafe (version control), and performance and analysis tools. Check out the Visual Studio.NET Feature Comparison at http://msdn.microsoft.com/vstudio/howtobuy/choosing.asp

Terminology

2.1 What is the CLI? Is it the same as the CLR?

The CLI (Common Language Infrastructure) is the definition of the fundamentals of the .NET framework - the Common Type System (CTS), metadata, the Virtual Execution Environment (VES) and its use of intermediate language (IL), and the support of multiple programming languages via the Common Language Specification (CLS). The CLI is documented through ECMA - see http://msdn.microsoft.com/net/ecma/ for more details.

The CLR (Common Language Runtime) is Microsoft's primary *implementation* of the CLI. Microsoft also have a shared source implementation known as ROTOR, for educational purposes, as well as the .NET Compact Framework for mobile devices. Non-Microsoft CLI implementations include Mono and DotGNU Portable. NET.

2.2 What is the CTS, and how does it relate to the CLS?

CTS = Common Type System. This is the full range of types that the .NET runtime understands. Not all .NET languages support all the types in the CTS.
CLS = Common Language Specification. This is a subset of the CTS which *all* .NET languages are expected to support. The idea is that any program which uses CLS-compliant types can interoperate with any .NET program written in any language. This interop is very fine-grained - for example a VB.NET class can inherit from a C# class.

2.3 What is IL?

IL = Intermediate Language. Also known as MSIL (Microsoft Intermediate Language) or CIL (Common Intermediate Language). All .NET source code (of any language) is compiled to IL during development. The IL is then converted to machine code at the point where the software is installed, or (more commonly) at run-time by a Just-In-Time (JIT) compiler.

2.4 What is C#?

C# is a new language designed by Microsoft to work with the .NET framework. In their "Introduction to C#" whitepaper, Microsoft describe C# as follows:

"C# is a simple, modern, object oriented, and type-safe programming language derived from C and C++. C# (pronounced “C sharp”) is firmly planted in the C and C++ family tree of languages, and will immediately be familiar to C and C++ programmers. C# aims to combine the high productivity of Visual Basic and the raw power of C++."

Substitute 'Java' for 'C#' in the quote above, and you'll see that the statement still works pretty well :-).

2.5 What does 'managed' mean in the .NET context?

The term 'managed' is the cause of much confusion. It is used in various places within .NET, meaning slightly different things.

Managed *code*: The .NET framework provides several core run-time services to the programs that run within it - for example exception handling and security. For these services to work, the code must provide a minimum level of information to the runtime. Such code is called managed code.

Managed *data*: This is data that is allocated and freed by the .NET runtime's garbage collector.

Managed *classes*: This is usually referred to in the context of Managed Extensions (ME) for C++. When using ME C++, a class can be marked with the __gc keyword. As the name suggests, this means that the memory for instances of the class is managed by the garbage collector, but it also means more than that. The class becomes a fully paid-up member of the .NET community with the benefits and restrictions that brings. An example of a benefit is proper interop with classes written in other languages - for example, a managed C++ class can inherit from a VB class. An example of a restriction is that a managed class can only inherit from one base class.

2.6 What is reflection?

All .NET compilers produce metadata about the types defined in the modules they produce. This metadata is packaged along with the module (modules in turn are packaged together in assemblies), and can be accessed by a mechanism called **reflection**. The System.Reflection namespace contains classes that can be used to interrogate the types for a module/assembly.

Using reflection to access .NET metadata is very similar to using ITypeLib/ITypeInfo to access type library data in COM, and it is used for similar purposes - e.g. determining data type sizes for marshaling data across context/process/machine boundaries.

Reflection can also be used to dynamically invoke methods (see System.Type.InvokeMember), or even create types dynamically at run-time (see System.Reflection.Emit.TypeBuilder).

3. Assemblies

3.1 What is an assembly?

An assembly is sometimes described as a logical .EXE or .DLL, and can be an *application* (with a main entry point) or a *library*. An assembly consists of one or more files (dlls, exes, html files etc), and represents a group of resources, type definitions, and implementations of those types. An assembly may also contain references to other assemblies. These resources, types and references are described in a block of data called a *manifest*. The manifest is part of the assembly, thus making the assembly self-describing.

An important aspect of assemblies is that they are part of the identity of a type. The identity of a type is the assembly that houses it combined with the type name. This means, for example, that if assembly A exports a type called T, and assembly B exports a type called T, the .NET runtime sees these as two completely different types. Furthermore, don't get confused between assemblies and namespaces - namespaces are merely a hierarchical way of organising type names. To the runtime, type names are type names, regardless of whether namespaces are used to organise the names. It's the

assembly plus the typename (regardless of whether the type name belongs to a namespace) that uniquely indentifies a type to the runtime.

Assemblies are also important in .NET with respect to security - many of the security restrictions are enforced at the assembly boundary. Finally, assemblies are the unit of versioning in .NET - more on this below.

3.2 How can I produce an assembly?

The simplest way to produce an assembly is directly from a .NET compiler. For example, the following C# program:

```
public class CTest
{
   public CTest() { System.Console.WriteLine( "Hello from CTest" ); }
}
```

can be compiled into a library assembly (dll) like this:

```
csc /t:library ctest.cs
```

You can then view the contents of the assembly by running the "IL Disassembler" tool that comes with the .NET SDK.

Alternatively you can compile your source into modules, and then combine the modules into an assembly using the assembly linker (al.exe). For the C# compiler, the /target:module switch is used to generate a module instead of an assembly.

3.3 What is the difference between a private assembly and a shared assembly?

- **Location and visibility**: A private assembly is normally used by a single application, and is stored in the application's directory, or a sub-directory beneath. A shared assembly is normally stored in the global assembly cache, which is a repository of assemblies maintained by the .NET runtime. Shared assemblies are usually libraries of code which many applications will find useful, e.g. the .NET framework classes.
- **Versioning**: The runtime enforces versioning constraints only on shared assemblies, not on private assemblies.

3.4 How do assemblies find each other?

By searching directory paths. There are several factors which can affect the path (such as the AppDomain host, and application configuration files), but for private assemblies the search path is normally the application's directory and its sub-directories. For shared assemblies, the search path is normally same as the private assembly path plus the shared assembly cache.

3.5 How does assembly versioning work?

Each assembly has a version number called the compatibility version. Also each reference to an assembly (from another assembly) includes both the name and version of the referenced assembly.

The version number has four numeric parts (e.g. 5.5.2.33). Assemblies with either of the first two parts different are normally viewed as incompatible. If the first two parts are the same, but the third is different, the assemblies are deemed as 'maybe compatible'. If only the fourth part is different, the assemblies are deemed compatible. However, this is just the default guideline - it is the version policy that decides to what extent these rules are enforced. The version policy can be specified via the application configuration file.

Remember: versioning is only applied to shared assemblies, not private assemblies.

3.6 How can I develop an application that automatically updates itself from the web?

4. Application Domains

4.1 What is an application domain?

An AppDomain can be thought of as a lightweight process. Multiple AppDomains can exist inside a Win32 process. The primary purpose of the AppDomain is to isolate applications from each other, and so it is particularly useful in hosting scenarios such as ASP.NET. An AppDomain can be destroyed by the host without affecting other AppDomains in the process.

Win32 processes provide isolation by having distinct memory address spaces. This is effective, but expensive. The .NET runtime enforces AppDomain isolation by keeping control over the use of memory - all memory in the AppDomain is managed by the .NET runtime, so the runtime can ensure that AppDomains do not access each other's memory.

One non-obvious use of AppDomains is for unloading types. Currently the only way to unload a .NET type is to destroy the AppDomain it is loaded into. This is particularly useful if you create and destroy types on-the-fly via reflection.

4.2 How does an AppDomain get created?

AppDomains are usually created by hosts. Examples of hosts are the Windows Shell, ASP.NET and IE. When you run a .NET application from the command-line, the host is the Shell. The Shell creates a new AppDomain for every application.

AppDomains can also be explicitly created by .NET applications. Here is a C# sample which creates an AppDomain, creates an instance of an object inside it, and then executes one of the object's methods:

```
using System;
using System.Runtime.Remoting;
using System.Reflection;

public class CAppDomainInfo : MarshalByRefObject
{
   public string GetName() { return AppDomain.CurrentDomain.FriendlyName; }
}

public class App
{
   public static int Main()
   {
      AppDomain ad = AppDomain.CreateDomain( "Andy's new domain" );
      CAppDomainInfo adInfo = (CAppDomainInfo)ad.CreateInstanceAndUnwrap(
         Assembly.GetCallingAssembly().GetName().Name, "CAppDomainInfo" );
      Console.WriteLine( "Created AppDomain name = " + adInfo.GetName() );
      return 0;
   }
}
```

4.3 Can I write my own .NET host?

Yes. For an example of how to do this, take a look at the source for the dm.net moniker developed by Jason Whittington and Don Box. There is also a code sample in the .NET SDK called CorHost.

5. Garbage Collection

5.1 What is garbage collection?

Garbage collection is a heap-management strategy where a run-time component takes responsibility for managing the lifetime of the memory used by objects. This concept is not new to .NET - Java and many other languages/runtimes have used garbage collection for some time.

5.2 Is it true that objects don't always get destroyed immediately when the last reference goes away?

Yes. The garbage collector offers no guarantees about the time when an object will be destroyed and its memory reclaimed.

There was an interesting thread on the DOTNET list, started by Chris Sells, about the implications of non-deterministic destruction of objects in C#. In October 2000, Microsoft's Brian Harry posted a lengthy analysis of the problem. Chris Sells' response to Brian's posting is here.

5.3 Why doesn't the .NET runtime offer deterministic destruction?

Because of the garbage collection algorithm. The .NET garbage collector works by periodically running through a list of all the objects that are currently being referenced by an application. All the objects that it doesn't find during this search are ready to be destroyed and the memory reclaimed. The implication of this algorithm is that the runtime doesn't get notified immediately when the final reference on an object goes away - it only finds out during the next 'sweep' of the heap.

Futhermore, this type of algorithm works best by performing the garbage collection sweep as rarely as possible. Normally heap exhaustion is the trigger for a collection sweep.

5.4 Is the lack of deterministic destruction in .NET a problem?

It's certainly an issue that affects component design. If you have objects that maintain expensive or scarce resources (e.g. database locks), you need to provide some way to tell the object to release the resource when it is done. Microsoft recommend that you provide a method called Dispose() for this purpose. However, this causes problems for distributed objects - in a distributed system who calls the Dispose() method? Some form of reference-counting or ownership-management mechanism is needed to handle distributed objects - unfortunately the runtime offers no help with this.

5.5 Should I implement Finalize on my class? Should I implement IDisposable?

This issue is a little more complex than it first appears. There are really two categories of class that require deterministic destruction - the first category manipulate unmanaged types directly, whereas the second category manipulate *managed* types that require deterministic destruction. An example of the first category is a class with an IntPtr member representing an OS file handle. An example of the second category is a class with a System.IO.FileStream member.

For the first category, it makes sense to implement IDisposable *and* override Finalize. This allows the object user to 'do the right thing' by calling Dispose, but also provides a fallback of freeing the unmanaged resource in the Finalizer, should the calling code fail in its duty. However this logic does not apply to the second category of class, with only managed resources. In this case implementing Finalize is pointless, as managed member objects cannot be accessed in the Finalizer. This is because there is no guarantee about the ordering of Finalizer execution. So only the Dispose method should be implemented. (If you think about it, it doesn't really make sense to call Dispose on member objects from a Finalizer anyway, as the member object's Finalizer will do the required cleanup.)
For classes that need to implement IDisposable *and* override Finalize, see Microsoft's documented pattern.

Note that some developers argue that implementing a Finalizer is always a bad idea, as it hides a bug in your code (i.e. the lack of a Dispose call). A less radical approach is to implement Finalize but include a Debug.Assert at the start, thus signalling the problem in developer builds but allowing the cleanup to occur in release builds.

5.6 Do I have any control over the garbage collection algorithm?

A little. For example the System.GC class exposes a Collect method, which forces the garbage collector to collect all unreferenced objects immediately.

Also there is a **gcConcurrent** setting that can be specified via the application configuration file. This specifies whether or not the garbage collector performs some of its collection activities on a separate thread. The setting only applies on multi-processor machines, and defaults to true.

5.7 How can I find out what the garbage collector is doing?

Lots of interesting statistics are exported from the .NET runtime via the '.NET CLR xxx' performance counters. Use Performance Monitor to view them.

5.8 What is the lapsed listener problem?

The lapsed listener problem is one of the primary causes of leaks in .NET applications. It occurs when a subscriber (or 'listener') signs up for a publisher's event, but fails to unsubscribe. The failure to unsubscribe means that the publisher maintains a reference to the subscriber as long as the publisher is alive. For some publishers, this may be the duration of the application.

This situation causes two problems. The obvious problem is the leakage of the subscriber object. The other problem is the performance degredation due to the publisher sending redundant notifications to 'zombie' subscribers.

There are at least a couple of solutions to the problem. The simplest is to make sure the subscriber is unsubscribed from the publisher, typically by adding an Unsubscribe() method to the subscriber. Another solution, documented here by Shawn Van Ness, is to change the publisher to use weak references in its subscriber list.

5.9 When do I need to use GC.KeepAlive?

It's very unintuitive, but the runtime can decide that an object is garbage much sooner than you expect. More specifically, an object can become garbage while a method is executing on the object, which is contrary to most developers' expectations. Chris Brumme explains the issue on his blog. I've taken Chris's code and expanded it into a full app that you can play with if you want to prove to yourself that this is a real problem:

```
using System;
using System.Runtime.InteropServices;

class Win32
{
   [DllImport("kernel32.dll")]
   public static extern IntPtr CreateEvent( IntPtr lpEventAttributes,
      bool bManualReset,bool bInitialState, string lpName);

   [DllImport("kernel32.dll", SetLastError=true)]
   public static extern bool CloseHandle(IntPtr hObject);

   [DllImport("kernel32.dll")]
   public static extern bool SetEvent(IntPtr hEvent);
}

class EventUser
{
   public EventUser()
   {
      hEvent = Win32.CreateEvent( IntPtr.Zero, false, false, null );
   }

   ~EventUser()
   {
      Win32.CloseHandle( hEvent );
      Console.WriteLine("EventUser finalized");
   }

   public void UseEvent()
   {
      UseEventInStatic( this.hEvent );
   }

   static void UseEventInStatic( IntPtr hEvent )
```

```
    {
        //GC.Collect();
        bool bSuccess = Win32.SetEvent( hEvent );
        Console.WriteLine( "SetEvent " + (bSuccess ? "succeeded" : "FAILED!") );
    }

    IntPtr hEvent;
}

class App
{
    static void Main(string[] args)
    {
        EventUser eventUser = new EventUser();
        eventUser.UseEvent();
    }
}
```

If you run this code, it'll probably work fine, and you'll get the following output:

```
SetEvent succeeded
EventDemo finalized
```

However, if you uncomment the GC.Collect() call in the UseEventInStatic() method, you'll get this output:

```
EventDemo finalized
SetEvent FAILED!
```

(Note that you need to use a release build to reproduce this problem.)

So what's happening here? Well, at the point where UseEvent() calls UseEventInStatic(), a copy is taken of the hEvent field, and there are no further references to the EventUser object anywhere in the code. So as far as the runtime is concerned, the EventUser object is garbage and can be collected. Normally of course the collection won't happen immediately, so you'll get away with it, but sooner or later a collection will occur at the wrong time, and your app will fail.

A solution to this problem is to add a call to GC.KeepAlive(this) to the end of the UseEvent method, as Chris explains.

6. Serialization

6.1 What is serialization?

Serialization is the process of converting an object into a stream of bytes. Deserialization is the opposite process, i.e. creating an object from a stream of bytes. Serialization/Deserialization is mostly used to transport objects (e.g. during remoting), or to persist objects (e.g. to a file or database).

6.2 Does the .NET Framework have in-built support for serialization?

There are two separate mechanisms provided by the .NET class library - XmlSerializer and SoapFormatter/BinaryFormatter. Microsoft uses XmlSerializer for Web Services, and SoapFormatter/BinaryFormatter for remoting. Both are available for use in your own code.

6.3 I want to serialize instances of my class. Should I use XmlSerializer, SoapFormatter or BinaryFormatter?

It depends. XmlSerializer has severe limitations such as the requirement that the target class has a parameterless constructor, and only public read/write properties and fields can be serialized. However, on the plus side, XmlSerializer has good support for customising the XML document that is produced or consumed. XmlSerializer's features mean that it is most suitable for cross-platform work, or for constructing objects from existing XML documents.

SoapFormatter and BinaryFormatter have fewer limitations than XmlSerializer. They can serialize private fields, for example. However they both require that the target class be marked with the [Serializable] attribute, so like XmlSerializer the class needs to be written with serialization in mind. Also there are some quirks to watch out for - for example on deserialization the constructor of the new object is not invoked.

The choice between SoapFormatter and BinaryFormatter depends on the application. BinaryFormatter makes sense where both serialization and deserialization will be performed on the .NET platform and where performance is important. SoapFormatter generally makes more sense in all other cases, for ease of debugging if nothing else.

6.4 Can I customise the serialization process?

Yes. XmlSerializer supports a range of attributes that can be used to configure serialization for a particular class. For example, a field or property can be marked with the [XmlIgnore] attribute to exclude it from serialization. Another example is the [XmlElement] attribute, which can be used to specify the XML element name to be used for a particular property or field.

Serialization via SoapFormatter/BinaryFormatter can also be controlled to some extent by attributes. For example, the [NonSerialized] attribute is the equivalent of XmlSerializer's [XmlIgnore] attribute. Ultimate control of the serialization process can be acheived by implementing the the ISerializable interface on the class whose instances are to be serialized.

6.5 Why is XmlSerializer so slow?

There is a once-per-process-per-type overhead with XmlSerializer. So the first time you serialize or deserialize an object of a given type in an application, there is a significant delay. This normally doesn't matter, but it may mean, for example, that XmlSerializer is a poor choice for loading configuration settings during startup of a GUI application.

6.6 Why do I get errors when I try to serialize a Hashtable?

XmlSerializer will refuse to serialize instances of any class that implements IDictionary, e.g. Hashtable. SoapFormatter and BinaryFormatter do not have this restriction.

6.7 XmlSerializer is throwing a generic "There was an error reflecting MyClass" error. How do I find out what the problem is?

Look at the InnerException property of the exception that is thrown to get a more specific error message.

6.8 Why am I getting an InvalidOperationException when I serialize an ArrayList?

XmlSerializer needs to know in advance what type of objects it will find in an ArrayList. To specify the type, use the XmlArrayItem attibute like this:

```
public class Person{
   public string Name;
   public int Age;
}

public class Population{
   [XmlArrayItem(typeof(Person))] public ArrayList People;
}
```

7. Attributes

7.1 What are attributes?

There are at least two types of .NET attribute. The first type I will refer to as a *metadata* attribute - it allows some data to be attached to a class or method. This data becomes part of the metadata for the class, and (like other class metadata) can be accessed via reflection. An example of a metadata attribute is [serializable], which can be attached to a class and means that instances of the class can be serialized.

```
[serializable] public class CTest {}
```

The other type of attribute is a *context* attribute. Context attributes use a similar syntax to metadata attributes but they are fundamentally different. Context attributes provide an interception mechanism whereby instance activation and method calls can be pre- and/or post-processed. If you have encountered Keith Brown's universal delegator you'll be familiar with this idea.

7.2 Can I create my own metadata attributes?

Yes. Simply derive a class from System.Attribute and mark it with the AttributeUsage attribute. For example:

```
[AttributeUsage(AttributeTargets.Class)]
public class InspiredByAttribute : System.Attribute {
   public string InspiredBy;
   public InspiredByAttribute( string inspiredBy ){
      InspiredBy = inspiredBy;
   }
}

[InspiredBy("Andy Mc's brilliant .NET FAQ")]
class CTest{
}
class CApp{
   public static void Main(){
      object[] atts = typeof(CTest).GetCustomAttributes(true);

      foreach( object att in atts )
         if( att is InspiredByAttribute )
            Console.WriteLine( "Class CTest was inspired by {0}",
((InspiredByAttribute)att).InspiredBy  );
   }
}
```

7.3 Can I create my own context attributes?

8. Code Access Security

8.1 What is Code Access Security (CAS)?

CAS is the part of the .NET security model that determines whether or not code is allowed to run, and what resources it can use when it is running. For example, it is CAS that will prevent a .NET web applet from formatting your hard disk.

8.2 How does CAS work?

The CAS security policy revolves around two key concepts - code groups and permissions. Each .NET assembly is a member of a particular code group, and each code group is granted the permissions specified in a **named permission set**.
For example, using the default security policy, a control downloaded from a web site belongs to the 'Zone - Internet' code group, which adheres to the permissions defined by the 'Internet' named permission set. (Naturally the 'Internet' named permission set represents a very restrictive range of permissions.)

8.3 Who defines the CAS code groups?

Microsoft defines some default ones, but you can modify these and even create your own. To see the code groups defined on your system, run 'caspol -lg' from the command-line. On my system it looks like this:

```
Level = Machine

Code Groups:

1. All code: Nothing
1.1. Zone - MyComputer: FullTrust
  1.1.1. Honor SkipVerification requests: SkipVerification
1.2. Zone - Intranet: LocalIntranet
1.3. Zone - Internet: Internet
1.4. Zone - Untrusted: Nothing
1.5. Zone - Trusted: Internet
1.6. StrongName -
00240000048000009400000006020000002400005253413100040000003

000000CFCB3291AA715FE99D40D49040336F9056D7886FED46775BC7BB5430BA4444FEF834
8EBD06
F962F39776AE4DC3B7B04A7FE6F49F25F740423EBF2C0B89698D8D08AC48D69CED0FC8F83B
465E08
07AC11EC1DCC7D054E807A43336DDE408A5393A48556123272CEEEE72F1660B71927D3856
1AABF5C
AC1DF1734633C602F8F2D5: Everything
```

Note the hierarchy of code groups - the top of the hierarchy is the most general ('All code'), which is then sub-divided into several groups, each of which in turn can be sub-divided. Also note that (somewhat counter-intuitively) a sub-group can be associated with a more permissive permission set than its parent.

8.4 How do I define my own code group?

Use caspol. For example, suppose you trust code from www.mydomain.com and you want it have full access to your system, but you want to keep the default restrictions for all other internet sites. To achieve this, you would add a new code group as a sub-group of the 'Zone - Internet' group, like this:

```
caspol -ag 1.3 -site www.mydomain.com FullTrust
```

Now if you run caspol -lg you will see that the new group has been added as group 1.3.1:

```
...
  1.3. Zone - Internet: Internet
    1.3.1. Site - www.mydomain.com: FullTrust
...
```

Note that the numeric label (1.3.1) is just a caspol invention to make the code groups easy to manipulate from the command-line. The underlying runtime never sees it.

8.5 How do I change the permission set for a code group?

Use caspol. If you are the machine administrator, you can operate at the 'machine' level - which means not only that the changes you make become the default for the machine, but also that users cannot change the permissions to be more permissive. If you are a normal (non-admin) user you can still modify the permissions, but only to make them more restrictive. For example, to allow intranet code to do what it likes you might do this:

```
caspol -cg 1.2 FullTrust
```

Note that because this is more permissive than the default policy (on a standard system), you should only do this at the machine level - doing it at the user level will have no effect.

8.6 Can I create my own permission set?

Yes. Use caspol -ap, specifying an XML file containing the permissions in the permission set. To save you some time, here is a sample file corresponding to the 'Everything' permission set - just edit to suit your needs. When you have edited the sample, add it to the range of available permission sets like this:

```
caspol -ap samplepermset.xml
```

```
<PermissionSet class="System.Security.NamedPermissionSet" version="1">
  - <Permission class="System.Security.Permissions.SecurityPermission,
       mscorlib, Ver=2000.14.1812.10, SN=03689116d3a4ae33"
       version="1">
      <Assertion />
      <UnmanagedCode />
      <Execution />
      <ControlThread />
      <ControlEvidence />
      <ControlPolicy />
      <SerializationFormatter />
      <ControlDomainPolicy />
      <ControlPrincipal />
    </Permission>
  - <Permission
       class="System.Security.Permissions.IsolatedStorageFilePermission,
       mscorlib, Ver=2000.14.1812.10, SN=03689116d3a4ae33"
       version="1">
      <Unrestricted />
    </Permission>
  - <Permission class="System.Security.Permissions.EnvironmentPermission,
       mscorlib, Ver=2000.14.1812.10, SN=03689116d3a4ae33"
       version="1">
      <Unrestricted />
    </Permission>
  - <Permission class="System.Security.Permissions.FileDialogPermission,
       mscorlib, Ver=2000.14.1812.10, SN=03689116d3a4ae33"
       version="1">
      <Unrestricted />
    </Permission>
  - <Permission class="System.Security.Permissions.FileIOPermission,
       mscorlib, Ver=2000.14.1812.10, SN=03689116d3a4ae33"
       version="1">
      <Unrestricted />
    </Permission>
  - <Permission class="System.Security.Permissions.ReflectionPermission,
       mscorlib, Ver=2000.14.1812.10, SN=03689116d3a4ae33"
       version="1">
      <Unrestricted />
    </Permission>
  - <Permission class="System.Security.Permissions.RegistryPermission,
       mscorlib, Ver=2000.14.1812.10, SN=03689116d3a4ae33"
       version="1">
      <Unrestricted />
    </Permission>
```

```
- <Permission class="System.Security.Permissions.UIPermission, mscorlib,
      Ver=2000.14.1812.10, SN=03689116d3a4ae33" version="1">
    <Unrestricted />
  </Permission>
  <Name>SamplePermSet</Name>
  <Description>By default this sample permission set is the same as the
    standard 'Everything' permission set - just edit to suit your
    needs.</Description>
</PermissionSet>
```

Then, to apply the permission set to a code group, do something like this:

```
caspol -cg 1.3 SamplePermSet
```

(By default, 1.3 is the 'Internet' code group)

8.7 I'm having some trouble with CAS. How can I troubleshoot the problem?

Caspol has a couple of options that might help. First, you can ask caspol to tell you what code group an assembly belongs to, using caspol -rsg. Similarly, you can ask what permissions are being applied to a particular assembly using caspol -rsp.

8.8 I can't be bothered with CAS. Can I turn it off?

Yes, as long as you are an administrator. Just run:

```
caspol -s off
```

9. Intermediate Language (IL)

9.1 Can I look at the IL for an assembly?

Yes. MS supply a tool called Ildasm that can be used to view the metadata and IL for an assembly.

9.2 Can source code be reverse-engineered from IL?

Yes, it is often relatively straightforward to regenerate high-level source from IL. Lutz Roeder's Reflector does a very good job of turning IL into C# or VB.NET.

9.3 How can I stop my code being reverse-engineered from IL?

You can buy an IL obfuscation tool. These tools work by 'optimising' the IL in such a way that reverse-engineering becomes much more difficult.

Of course if you are writing web services then reverse-engineering is not a problem as clients do not have access to your IL.

9.4 Can I write IL programs directly?

Yes. Peter Drayton posted this simple example to the DOTNET mailing list:

```
.assembly MyAssembly {}
.class MyApp {
  .method static void Main() {
    .entrypoint
    ldstr      "Hello, IL!"
    call       void System.Console::WriteLine(class System.Object)
    ret
  }
}
```

Just put this into a file called hello.il, and then run ilasm hello.il. An exe assembly will be generated.

9.5 Can I do things in IL that I can't do in C#?

Yes. A couple of simple examples are that you can throw exceptions that are not derived from System.Exception, and you can have non-zero-based arrays.

10. Implications for COM

10.1 Does .NET replace COM?

This subject causes a lot of controversy, as you'll see if you read the mailing list archives. Take a look at the following two threads:

http://discuss.develop.com/archives/wa.exe?A2=ind0007&L=DOTNET&D=0&P=68241
http://discuss.develop.com/archives/wa.exe?A2=ind0007&L=DOTNET&P=R60761

COM is about introducing type into components, period. The primary theme of COM was that we loaded code based on types (CoCreateInstance) and that we resolved entry points based on types (QueryInterface). This was a big advance from the days when we loaded code based on files (LoadLibrary) and
resolved entry points based on flat symbolic names (GetProcAddress). Nothing in CLR changes that. The context architecture of MTS made this loader extensible, allowing aspects of your implementation to be expressed via declarative attributes as well as executable statements.
Where classic COM has always fallen short was in the area of runtime type information. In classic COM, the TLB describes the EXPORTED types only. IMPORTED types are opaque, as are INTERNAL types used within the component boundary. The former makes it impossible to perform dependency analysis for deployment, versioning, etc. The latter makes it impossible for certain classes of services to do anything useful without massive programmer intervention. Thankfully, CLR provides the system with "perfect" type information, which as you well know, enables tons of goodness from both MS and third parties.

Focusing on things like GC or IL/JIT performance is really a red herring, and it looks like you have avoided that trap. Your comment, however, really cuts to the chase in terms of the "soul of COM" - that is, the role of the interface.

In classic COM, all object references were interface-based. In CLR, object references can be either interface-based or class-based. Yes, many of the folks at MS now advocate using classes in many of the scenarios where interfaces were used in the past. In the intra-component case where incremental deployment is a non-issue, this shouldn't seem so radical, as most of today's COM components use class-based references internally (most ATL programmers work this way, I know I do). In fact, I am personally glad that the same technology I use for loading cross-component types is used for intra-component types. This solves tons of the old VB "New vs. CreateObject" bugs that today's COM programmers still battle.

For inter-component sharing of class-based references, the water is more murky. On the one hand, if you look at passing class-based references as parameters, your aren't that far off from passing structs, and as long as you stick to sealed, MBV, class definitions that are immutable, this should be pretty easy to swallow. The more problematic case is where abstract classes are used in lieu of interfaces. Personally, I am still skeptical about this one, but I see (but don't necessarily agree with) the arguments in favor of the approach. As is always the case, the community at large figures our which parts of a programming model make sense and which don't, so I am willing to be proven wrong. As far as I can tell, none of the ideas worth keeping from the IUnknown era have been dropped. Rather, the big four concepts of COM (contexts, attributes, classes and interfaces) simply have a better supporting implementation called CLR, which until not that long ago was called "COM+" (run REGASM.exe and look up your code in the registry ;-)).

So, is CLR COM? If you think COM is IUnknown and a set of APIs, then the answer is no*. If you think COM is a programming model based on typed components, interfaces, attributes and context, then the answer is yes. I fall firmly in the latter camp, so to me, CLR is just better plumbing for the same programming model I've spent years working with.

The bottom line is that .NET has its own mechanisms for type interaction, and they don't use COM. No IUnknown, no IDL, no typelibs, no registry-based activation. This is mostly good, as a lot of COM was ugly. Generally speaking, .NET allows you to package and use components in a similar way to COM, but makes the whole thing a bit easier.

10.2 Is DCOM dead?

Pretty much, for .NET developers. The .NET Framework has a new remoting model which is not based on DCOM. DCOM was pretty much dead anyway, once firewalls became widespread and Microsoft got SOAP fever. Of course DCOM will still be used in interop scenarios.

10.3 Is COM+ dead?

Not immediately. The approach for .NET 1.0 was to provide access to the existing COM+ services (through an interop layer) rather than replace the services with native .NET ones. Various tools and attributes were provided to make this as painless as possible. Over time it is expected that interop will become more seamless - this may mean that some services become a core part of the CLR, and/or it may mean that some services will be rewritten as managed code which runs on top of the CLR.

For more on this topic, search for postings by Joe Long in the archives - Joe is the MS group manager for COM+. Start with this message:

http://discuss.develop.com/archives/wa.exe?A2=ind0007&L=DOTNET&P=R68370

10.4 Can I use COM components from .NET programs?

Yes. COM components are accessed from the .NET runtime via a Runtime Callable Wrapper (RCW). This wrapper turns the COM interfaces exposed by the COM component into .NET-compatible interfaces. For oleautomation interfaces, the RCW can be generated automatically from a type library. For non-oleautomation interfaces, it may be necessary to develop a custom RCW which manually maps the types exposed by the COM interface to .NET-compatible types.

Here's a simple example for those familiar with ATL. First, create an ATL component which implements the following IDL:

```
import "oaidl.idl";
import "ocidl.idl";

[
   object,
   uuid(EA013F93-487A-4403-86EC-FD9FEE5E6206),
   helpstring("ICppName Interface"),
   pointer_default(unique),
   oleautomation
]

interface ICppName : IUnknown
{
   [helpstring("method SetName")] HRESULT SetName([in] BSTR name);
   [helpstring("method GetName")] HRESULT GetName([out,retval] BSTR *pName );
};

[
   uuid(F5E4C61D-D93A-4295-A4B4-2453D4A4484D),
   version(1.0),
   helpstring("cppcomserver 1.0 Type Library")
]
library CPPCOMSERVERLib
{
   importlib("stdole32.tlb");
   importlib("stdole2.tlb");
   [
      uuid(600CE6D9-5ED7-4B4D-BB49-E8D5D5096F70),
      helpstring("CppName Class")
   ]
   coclass CppName
   {
      [default] interface ICppName;
   };
};
```

When you've built the component, you should get a typelibrary. Run the TLBIMP utility on the typelibary, like this:

```
tlbimp cppcomserver.tlb
```

If successful, you will get a message like this:

```
Typelib imported successfully to CPPCOMSERVERLib.dll
```

You now need a .NET client - let's use C#. Create a .cs file containing the following code:

```
using System;
using CPPCOMSERVERLib;

public class MainApp
{
   static public void Main()
   {
      CppName cppname = new CppName();
      cppname.SetName( "bob" );
      Console.WriteLine( "Name is " + cppname.GetName() );
   }
}
```

Compile the C# code like this:

```
csc /r:cppcomserverlib.dll csharpcomclient.cs
```

Note that the compiler is being told to reference the DLL we previously generated from the typelibrary using TLBIMP. You should now be able to run csharpcomclient.exe, and get the following output on the console:

```
Name is bob
```

10.5 Can I use .NET components from COM programs?

Yes. .NET components are accessed from COM via a COM Callable Wrapper (CCW). This is similar to a RCW (see previous question), but works in the opposite direction. Again, if the wrapper cannot be automatically generated by the .NET development tools, or if the automatic behaviour is not desirable, a custom CCW can be developed. Also, for COM to 'see' the .NET component, the .NET component must be registered in the registry.

Here's a simple example. Create a C# file called testcomserver.cs and put the following in it:

```
using System;
using System.Runtime.InteropServices;

namespace AndyMc
{
   [ClassInterface(ClassInterfaceType.AutoDual)]
   public class CSharpCOMServer
   {
      public CSharpCOMServer() {}
      public void SetName( string name ) { m_name = name; }
      public string GetName() { return m_name; }
      private string m_name;
   }
}
```

Then compile the .cs file as follows:

```
csc /target:library testcomserver.cs
```

You should get a dll, which you register like this:

```
regasm testcomserver.dll /tlb:testcomserver.tlb /codebase
```

Now you need to create a client to test your .NET COM component. VBScript will do - put the following in a file called comclient.vbs:

```
Dim dotNetObj
Set dotNetObj = CreateObject("AndyMc.CSharpCOMServer")
dotNetObj.SetName ("bob")
MsgBox "Name is " & dotNetObj.GetName()
```

and run the script like this:

```
wscript comclient.vbs
```

And hey presto you should get a message box displayed with the text "Name is bob".

An alternative to the approach above it to use the dm.net moniker developed by Jason Whittington and Don Box.

10.6 Is ATL redundant in the .NET world?

Yes. ATL will continue to be valuable for writing COM components for some time, but it has no place in the .NET world.

11. Miscellaneous

11.1 How does .NET remoting work?

.NET remoting involves sending messages along channels. Two of the standard channels are HTTP and TCP. TCP is intended for LANs only - HTTP can be used for LANs or WANs (internet).

Support is provided for multiple message serializarion formats. Examples are SOAP (XML-based) and binary. By default, the HTTP channel uses SOAP (via the .NET runtime Serialization SOAP Formatter), and the TCP channel uses binary (via the .NET runtime Serialization Binary Formatter). But either channel can use either serialization format.

There are a number of styles of remote access:

- **SingleCall**. Each incoming request from a client is serviced by a new object. The object is thrown away when the request has finished.
- **Singleton**. All incoming requests from clients are processed by a single server object.

- **Client-activated object**. This is the old stateful (D)COM model whereby the client receives a reference to the remote object and holds that reference (thus keeping the remote object alive) until it is finished with it.

Distributed garbage collection of objects is managed by a system called 'leased based lifetime'. Each object has a lease time, and when that time expires the object is disconnected from the .NET runtime remoting infrastructure. Objects have a default renew time - the lease is renewed when a successful call is made from the client to the object. The client can also explicitly renew the lease.

If you're interested in using XML-RPC as an alternative to SOAP, take a look at Charles Cook's XML-RPC.Net.

11.2 How can I get at the Win32 API from a .NET program?

Use P/Invoke. This uses similar technology to COM Interop, but is used to access static DLL entry points instead of COM objects. Here is an example of C# calling the Win32 MessageBox function:

```
using System;
using System.Runtime.InteropServices;

class MainApp
{
   [DllImport("user32.dll", EntryPoint="MessageBox", SetLastError=true,
CharSet=CharSet.Auto)]
   public static extern int MessageBox(int hWnd, String strMessage, String strCaption, uint
uiType);

   public static void Main()
   {
      MessageBox( 0, "Hello, this is PInvoke in operation!", ".NET", 0 );
   }
}
```

Pinvoke.net is a great resource for off-the-shelf P/Invoke signatures.

11.3 How do I write to the application configuration file at runtime?

Why Writing Into .NET Application Configuration Files Is a Bad Idea -

I seem to write about this often enough in news groups and mailing lists that I thought I'd put it up here - that way I can just post a link next time...

.NET applications can have a configuration file associated with them. In web apps, the file is called web.config. For a normal Windows or console application, it is has the same name as the executable with .config tacked on

the end. So if your app is called Foo.exe, you would put the configuration file in the same directory and call it Foo.exe.config. In Visual Studio .NET projects, the file will be called App.config - VS.NET copies it into your build output directory and changes the name appropriately.

.NET itself looks for certain settings inside this file. For example, the assembly resolver can be configured this way. It can also be used to store some application-specific settings in its <appSettings> element. Because of this, a frequently asked question emerges:

How do I write settings into the application configuration file at runtime?

The short answer is: **"Don't do that."**

The slightly more informative but still fairly succinct answer is: "You quite often can't, and even when you can you don't want to."

Here's the long answer.

Why the Config File Might Not Be Writable

The configuration file is stored in the same directory as the executable itself. It's pretty common for users not to have permission to write to this directory. Indeed that's just common sense and good practice - ensuring that the user doesn't have permission to write into directories where executables are stored prevents a lot of viruses, spyware, and other malware from wreaking havoc.

This is why the Program Files directory is configured not to allow normal users to write to it.

Unfortunately, not everyone runs as a normal user - lots of people run as administrators on their Windows machines. This is a shame as it means they are throwing away a lot of the protection Windows can offer them. Of course one of the reasons so many people run as administrators is because of crappy applications that attempt to write into their installation directories.

Don't make your application one of those crappy applications. Don't try and write into the directory where your application is installed, because if you do that, your program won't run properly unless the user is an administrator.

Of course there may be other reasons that the file isn't writable. If your application is deployed via HTTP, the configuration file will live on the web server, and you are unlikely to be able to modify it.

Why You Wouldn't Want to Write Into It Even If You Could

Let's suppose you've ignored the above, and have decided to write a crappy application that writes into the directory it runs from. (Welcome to Keith Brown's Hall of Shame by the way.) You still don't want to be writing user settings in there. (And there's no need to put application-wide settings there either.)

Lots of machines have multiple users. And not just terminal servers - all my home machines have multiple users because I'm not the only person who logs into them. They don't tend to have multiple simultaneous users, but that's not the point. I don't want someone else's configuration choices to affect my account.

For example, I have a separate login I use on my laptop for doing presentations. All of the fonts are configured to be extra large and legible. The resolution is set for a projector rather than the laptop's own screen. It would be hugely annoying if these things were configured on a per-machine basis, because I'd have to keep reconfiguring them, rather than simply being able to switch user accounts. (Actually ILDASM stores font settings in a machine-wide way, and it really bugs me.)

If you store user settings in the application configuration file, you will be storing one load of settings that apply to all users on the machine. Of course you could fix that by rolling your own mechanism to keep each user's settings isolated. But you'd be mad to build such a thing when there's already a perfectly good system for doing this built into Windows!

Where Should I Put This Information Then?

The user's profile directory is the appropriate place for user settings. Since there is a profile directory for each user, this keeps the settings separate. Moreover, if the profile is configured as a roaming profile, the user's settings will then follow them around the network as they log into various machines. And if they back up their Documents and Settings directory, your application's settings will be backed up as a part of that operation. What's not to like?

So how do you locate this directory? .NET makes that very easy, thanks to the System.Environment class:

```
Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData)
```

That will return a string, containing the path of the Application Data subdirectory of the user's profile. (If you want to store data that does not

become part of the roaming profile, use LocalApplicationData instead.) You should create a subdirectory named after your company, and inside that a subdirectory named after your program. (If you look inside your profile's Application Data directory, you'll see that this is the structure applications usually use.) You may also choose to put a version-specific subdirectory underneath the application name directory.

(If your code might need to run with partial trust, look at the Isolated Storage APIs. These are a bit more cumbersome than just creating files in the right directory, and create some funny-looking paths. But they have the considerable advantage of working in partial trust scenarios.)

Note that even if you want to store application-wide settings, you still don't have to write into the application installation directory. Just do this:

```
Environment.GetFolderPath(Environment.SpecialFolder.CommonApplicationData)
```

That returns a path for machine-wide settings. Not all users will have write access to this by the way, so be careful.

So What Exactly Is The Configuration File For Then?

The configuration file is really only for settings configured at deployment time. It can be used to deal with versioning issues with .NET components. And it's often used for connections strings - it's useful to be able to deploy an application to connect to a test or staging server, but this is not something you'd normally change in production once the application is deployed.

11.4 What is the difference between an event and a delegate?

An event is just a wrapper for a multicast delegate. Adding a public event to a class is almost the same as adding a public multicast delegate field. In both cases, subscriber objects can register for notifications, and in both cases the publisher object can send notifications to the subscribers. However, a public multicast delegate has the undesirable property that external objects can *invoke* the delegate, something we'd normally want to restrict to the publisher. Hence events - an event adds public methods to the containing class to add and remove receivers, but does not make the invocation mechanism public.

11.5 What size is a .NET object?

Each instance of a reference type has two fields maintained by the runtime - a method table pointer and a sync block. These are 4 bytes each on a 32-bit system, making a total of 8 bytes per object overhead. Obviously the instance data for the type must be added to this to get the overall size of the object. So, for example, instances of the following class are 12 bytes each:

```
class MyInt
{
   ...
   private int x;
}
```

However, note that with the current implementation of the CLR there seems to be a minimum object size of 12 bytes, even for classes with no data (e.g. System.Object).

Values types have no equivalent overhead.

12. .NET 2.0

12.1 What are the new features of .NET 2.0?

Generics, anonymous methods, partial classes, iterators, property visibility (separate visibility for get and set) and static classes. See http://msdn.microsoft.com/msdnmag/issues/04/05/C20/default.aspx for more information about these features.

12.2 What are the new 2.0 features useful for?

Generics are useful for writing efficient type-independent code, particularly where the types might include value types. The obvious application is container classes, and the .NET 2.0 class library includes a suite of generic container classes in the System.Collections.Generic namespace. Here's a simple example of a generic container class being used:

```
List<int> myList = new List<int>();
myList.Add( 10 );
```

Anonymous methods reduce the amount of code you have to write when using delegates, and are therefore especially useful for GUI programming. Here's an example

```
AppDomain.CurrentDomain.ProcessExit += delegate { Console.WriteLine("Process ending ..."); };
```

Partial classes is a useful feature for separating machine-generated code from hand-written code in the same class, and will therefore be heavily used by development tools such as Visual Studio.

Iterators reduce the amount of code you need to write to implement IEnumerable/IEnumerator. Here's some sample code:

```
static void Main()
{
   RandomEnumerator re = new RandomEnumerator( 5 );
   foreach( double r in re )
      Console.WriteLine( r );
   Console.Read();
}

class RandomEnumerator : IEnumerable<double>
{
   public RandomEnumerator(int size) { m_size = size; }

   public IEnumerator<double> GetEnumerator()
   {
      Random rand = new Random();
      for( int i=0; i < m_size; i++ )
         yield return rand.NextDouble();
   }

   int m_size = 0;
}
```

The use of 'yield return' is rather strange at first sight. It effectively synthethises an implementation of IEnumerator, something we had to do manually in .NET 1.x.

12.3 What's the problem with .NET generics?

.NET generics work great for container classes. But what about other uses? Well, it turns out that .NET generics have a major limitation - they require the type parameter to be *constrained*. For example, you cannot do this:

```
static class Disposer<T>
{
   public static void Dispose(T obj) { obj.Dispose(); }
}
```

The C# compiler will refuse to compile this code, as the type T has not been constrained, and therefore only supports the methods of System.Object. Dispose is not a method on System.Object, so the compilation fails. To fix this code, we need to add a **where** clause, to reassure the compiler that our type T does indeed have a Dispose method

```
static class Disposer<T> where T : IDisposable
{
   public static void Dispose(T obj) { obj.Dispose(); }
}
```

The problem is that the requirement for explicit contraints is very limiting. We can use constraints to say that T implements a particular *interface*, but we can't dilute that to simply say that T implements a particular *method*. Contrast this with C++ templates (for example), where no constraint at all is required - it is *assumed* (and verified at compile time) that if the code

invokes the Dispose() method on a type, then the type will support the method.

In fact, after writing generic code with interface constraints, we quickly see that we haven't gained much over non-generic interface-based programming. For example, we can easily rewrite the Disposer class without generics:

```
static class Disposer
{
   public static void Dispose( IDisposable obj ) { obj.Dispose(); }
}
```

12.4 What's new in the .NET 2.0 class library?

Here is a selection of new features in the .NET 2.0 class library (beta 1):

- Generic collections in the System.Collections.Generic namespace.
- The **System.Nullable<T>** type. (Note that C# has special syntax for this type, e.g. int? is equivalent to Nullable<int>)
- The **GZipStream** and **DeflateStream** classes in the System.IO.Compression namespace.
- The **Semaphore** class in the System.Threading namespace.
- Wrappers for DPAPI in the form of the **ProtectedData** and **ProtectedMemory** classes in the System.Security.Cryptography namespace.
- The IPC remoting channel in the System.Runtime.Remoting.Channels.Ipc namespace, for optimised intra-machine communication.

1) The C# keyword .int. maps to which .NET type?

1. System.Int16
2. **System.Int32**
3. System.Int64
4. System.Int128

2) Which of these string definitions will prevent escaping on backslashes in C#?

1. string s = #.n Test string.;
2. string s = ..n Test string.;
3. **string s = @.n Test string.;**
4. string s = .n Test string.;

3) Which of these statements correctly declares a two-dimensional array in C#?

1. **int[,] myArray;**
2. int[][] myArray;
3. int[2] myArray;
4. System.Array[2] myArray;

4) If a method is marked as protected internal who can access it?

1. Classes that are both in the same assembly and derived from the declaring class.
2. Only methods that are in the same class as the method in question.
3. Internal methods can be only be called using reflection.
4. **Classes within the same assembly, and classes derived from the declaring class.**

5) What is boxing?

a) Encapsulating an object in a value type.

b) Encapsulating a copy of an object in a value type.

c) Encapsulating a value type in an object.

d) **Encapsulating a copy of a value type in an object.**

6) What compiler switch creates an xml file from the xml comments in the files in an assembly?

1. /text
2. **/doc**
3. /xml
4. /help

7) What is a satellite Assembly?

1. A peripheral assembly designed to monitor permissions requests from an application.
2. Any DLL file used by an EXE file.
3. **An assembly containing localized resources for another assembly.**
4. An assembly designed to alter the appearance or .skin. of an application.

8) What is a delegate?

1. **A strongly typed function pointer.**
2. A light weight thread or process that can call a single method.
3. A reference to an object in a different process.
4. An inter-process message channel.

9) How does assembly versioning in .NET prevent DLL Hell?

1. The runtime checks to see that only one version of an assembly is on the machine at any one time.
2. **.NET allows assemblies to specify the name AND the version of any assemblies they need to run.**
3. The compiler offers compile time checking for backward compatibility.
4. It doesn.t.

10) Which .Gang of Four. design pattern is shown below?

```
public class A {
   private A instance;
   private A() {
   }

   public
static A Instance {
      get {
         if ( A == null )
            A = new A();

         return instance;

      }

   }

}
```

1. Factory
2. Abstract Factory
3. **Singleton**
4. Builder

11) In the NUnit test framework, which attribute must adorn a test class in order for it to be picked up by the NUnit GUI?

1. TestAttribute
2. TestClassAttribute
3. **TestFixtureAttribute**
4. NUnitTestClassAttribute

12) Which of the following operations can you NOT perform on an ADO.NET DataSet?

1. A DataSet can be synchronised with the database.
2. **A DataSet can be synchronised with a RecordSet.**
3. A DataSet can be converted to XML.
4. You can infer the schema from a DataSet.

13) In Object Oriented Programming, how would you describe encapsulation?

1. The conversion of one type of object to another.
2. The runtime resolution of method calls.
3. The exposition of data.
4. **The separation of interface and implementation.**

.NET deployment questions

1. **What do you know about .NET assemblies?** Assemblies are the smallest units of versioning and deployment in the .NET application. Assemblies are also the building blocks for programs such as Web services, Windows services, serviced components, and .NET remoting applications.
2. **What's the difference between private and shared assembly?** Private assembly is used inside an application only and does not have to be identified by a strong name. Shared assembly can be used by multiple applications and has to have a strong name.
3. **What's a strong name?** A strong name includes the name of the assembly, version number, culture identity, and a public key token.
4. **How can you tell the application to look for assemblies at the locations other than its own install?** Use the
   directive in the XML .config file for a given application.

   <probing privatePath="c:\mylibs; bin\debug" />

   should do the trick. Or you can add additional search paths in the Properties box of the deployed application.

5. **How can you debug failed assembly binds?** Use the Assembly Binding Log Viewer (fuslogvw.exe) to find out the paths searched.
6. **Where are shared assemblies stored?** Global assembly cache.
7. **How can you create a strong name for a .NET assembly?** With the help of Strong Name tool (sn.exe).
8. **Where's global assembly cache located on the system?** Usually C:\winnt\assembly or C:\windows\assembly.
9. **Can you have two files with the same file name in GAC?** Yes, remember that GAC is a very special folder, and while normally you

would not be able to place two files with the same name into a Windows folder, GAC differentiates by version number as well, so it's possible for MyApp.dll and MyApp.dll to co-exist in GAC if the first one is version 1.0.0.0 and the second one is 1.1.0.0.

10. **So let's say I have an application that uses MyApp.dll assembly, version 1.0.0.0. There is a security bug in that assembly, and I publish the patch, issuing it under name MyApp.dll 1.1.0.0. How do I tell the client applications that are already installed to start using this new MyApp.dll?** Use publisher policy. To configure a publisher policy, use the publisher policy configuration file, which uses a format similar app .config file. But unlike the app .config file, a publisher policy file needs to be compiled into an assembly and placed in the GAC.
11. **What is delay signing?** Delay signing allows you to place a shared assembly in the GAC by signing the assembly with just the public key. This allows the assembly to be signed with the private key at a later stage, when the development process is complete and the component or assembly is ready to be deployed. This process enables developers to work with shared assemblies **as if** they were strongly named, and it secures the private key of the signature from being accessed at different stages of development.

13. Class Library

13.1 Threads

13.1.1 How do I spawn a thread?

Create an instance of a System.Threading.Thread object, passing it an instance of a ThreadStart delegate that will be executed on the new thread. For example:

```
class MyThread
{
   public MyThread( string initData )
   {
      m_data = initData;
      m_thread = new Thread( new ThreadStart(ThreadMain) );
      m_thread.Start();
   }

   // ThreadMain() is executed on the new thread.
   private void ThreadMain()
   {
      Console.WriteLine( m_data );
   }

   public void WaitUntilFinished()
   {
      m_thread.Join();
   }
```

```
    private Thread m_thread;
    private string m_data;
}
```

In this case creating an instance of the MyThread class is sufficient to spawn the thread and execute the MyThread.ThreadMain() method:

```
MyThread t = new MyThread( "Hello, world." );
t.WaitUntilFinished();
```

13.1.2 How do I stop a thread?

There are several options. First, you can use your own communication mechanism to tell the ThreadStart method to finish. Alternatively the Thread class has in-built support for instructing the thread to stop. The two principle methods are Thread.Interrupt() and Thread.Abort(). The former will cause a ThreadInterruptedException to be thrown on the thread when it next goes into a WaitJoinSleep state. In other words, Thread.Interrupt is a polite way of asking the thread to stop when it is no longer doing any useful work. In contrast, Thread.Abort() throws a ThreadAbortException regardless of what the thread is doing. Furthermore, the ThreadAbortException cannot normally be caught (though the ThreadStart's finally method will be executed). Thread.Abort() is a heavy-handed mechanism which should not normally be required.

13.1.3 How do I use the thread pool?

By passing an instance of a WaitCallback delegate to the ThreadPool.QueueUserWorkItem() method

```
class CApp
{
    static void Main(){
        string s = "Hello, World";
        ThreadPool.QueueUserWorkItem( new WaitCallback( DoWork ), s );

        Thread.Sleep( 1000 );    // Give time for work item to be executed
    }

    // DoWork is executed on a thread from the thread pool.
    static void DoWork( object state ){
        Console.WriteLine( state );
    }
}
```

13.1.4 How do I know when my thread pool work item has completed?

There is no way to query the thread pool for this information. You must put code into the WaitCallback method to signal that it has completed. Events are useful for this.

13.1.5 How do I prevent concurrent access to my data?

Each object has a concurrency lock (critical section) associated with it. The System.Threading.Monitor.Enter/Exit methods are used to acquire and release this lock. For example, instances of the following class only allow one thread at a time to enter method f():

```
class C{
   public void f(){
      try{
         Monitor.Enter(this);
         ...
      }
      finally{
         Monitor.Exit(this);
      }
   }
}
```

C# has a 'lock' keyword which provides a convenient shorthand for the code above:

```
class C{
   public void f(){
      lock(this){
         ...
      }
   }
}
```

Note that calling Monitor.Enter(myObject) does NOT mean that all access to myObject is serialized. It means that the synchronisation lock associated with myObject has been acquired, and no other thread can acquire that lock until Monitor.Exit(o) is called. In other words, this class is functionally equivalent to the classes above:

```
class C{
   public void f(){
      lock( m_object ){
         ...
      }
   }

   private m_object = new object();
}
```

Actually, it could be argued that this version of the code is superior, as the lock is totally encapsulated within the class, and not accessible to the user of the object.

13.1.6 Should I use ReaderWriterLock instead of Monitor.Enter/Exit?

Maybe, but be careful. ReaderWriterLock is used to allow multiple threads to read from a data source, while still granting exclusive access to a single writer thread. This makes sense for data access that is mostly read-only, but there are some caveats. First, ReaderWriterLock is relatively poor performing compared to Monitor.Enter/Exit, which offsets some of the benefits. Second, you need to be very sure that the data structures you are accessing fully support multithreaded read access. Finally, there is apparently a bug in the v1.1 ReaderWriterLock that can cause starvation for writers when there are a large number of readers.

13.2 Tracing

13.2.1 Is there built-in support for tracing/logging?

Yes, in the System.Diagnostics namespace. There are two main classes that deal with tracing - Debug and Trace. They both work in a similar way - the difference is that tracing from the Debug class only works in builds that have the DEBUG symbol defined, whereas tracing from the Trace class only works in builds that have the TRACE symbol defined. Typically this means that you should use System.Diagnostics.Trace.WriteLine for tracing that you want to work in debug and release builds, and System.Diagnostics.Debug.WriteLine for tracing that you want to work only in debug builds.

13.2.2 Can I redirect tracing to a file?

Yes. The Debug and Trace classes both have a Listeners property, which is a collection of sinks that receive the tracing that you send via Debug.WriteLine and Trace.WriteLine respectively. By default the Listeners collection contains a single sink, which is an instance of the DefaultTraceListener class. This sends output to the Win32 OutputDebugString() function and also the System.Diagnostics.Debugger.Log() method. This is useful when debugging, but if you're trying to trace a problem at a customer site, redirecting the output to a file is more appropriate. Fortunately, the TextWriterTraceListener class is provided for this purpose.

Here's how to use the TextWriterTraceListener class to redirect Trace output to a file:

```
Trace.Listeners.Clear();
FileStream fs = new FileStream( @"c:\log.txt", FileMode.Create, FileAccess.Write );
Trace.Listeners.Add( new TextWriterTraceListener( fs ) );
```

```
Trace.WriteLine( @"This will be writen to c:\log.txt!" );
Trace.Flush();
```

Note the use of Trace.Listeners.Clear() to remove the default listener. If you don't do this, the output will go to the file *and* OutputDebugString(). Typically this is not what you want, because OutputDebugString() imposes a big performance hit.

13.2.3 Can I customise the trace output?

Yes. You can write your own TraceListener-derived class, and direct all output through it. Here's a simple example, which derives from TextWriterTraceListener (and therefore has in-built support for writing to files, as shown above) and adds timing information and the thread ID for each trace line:

```
class MyListener : TextWriterTraceListener
{
   public MyListener( Stream s ) : base(s)
   {
   }

   public override void WriteLine( string s )
   {
      Writer.WriteLine( "{0:D8} [{1:D4}] {2}",
         Environment.TickCount - m_startTickCount,
         AppDomain.GetCurrentThreadId(),
         s );
   }

   protected int m_startTickCount = Environment.TickCount;
}
```

(Note that this implementation is not complete - the TraceListener.Write method is not overridden for example.)

The beauty of this approach is that when an instance of MyListener is added to the Trace.Listeners collection, all calls to Trace.WriteLine() go through MyListener, including calls made by referenced assemblies that know nothing about the MyListener class.

13.2.4 Are there any third party logging components available?

Log4net is a port of the established log4j Java logging component. log4net is a tool to help the programmer output log statements to a variety of output targets. log4net is a port of the excellent log4j framework to the .NET runtime. We have kept the framework similar in spirit to the original log4j while taking advantage of new features in the .NET runtime. For more information on log4net see the features document.

log4net is part of the Apache Logging Services project. The Logging Services project is intended to provide cross-language logging services for purposes of application debugging and auditing.

1. Explain dotnet framework ?
   The dot net Framework has two main components CLR and .NET Libraries. CLR (common language runtimes), that actually runs the code manages so many things for example code execution, garbage collection, memory allocation, thread management etc. Apart from CLR, the .NET framework contains .NET libraries, which are collection of namespaces and classes. The classes and namespaces are kept in a systematic way and can be used in making any application, code reuability etc. The root namespace of .NET framework is System, with this namespace many namespaces like web (system.web), data (system.data), windows (system.windows) are generated which can be further have their namespaces.

   2. What is the difference between Metadata and Menifest ?
   Menifest descriubes the assembely itself. Assembely name, version number, culture information. strong name, list of all files, type reference and reference assembely. While the Metadata describes the contents within the assembely. like classes, interfaces, namespaces, base class, scope, properties and their parameters etc.

   3. What are public and private assemblies ? differences and scope ?
   Public assembly are the dll/exe file that can be used in different application. The main advantage of public assemblies is code reusability. These can be used in different machine on different computers. These are also called as shared assemblies. Private assembly is the assembelyinfo.cs or assembelyinfo.vb file within an application. An application must have one private assembely, outside this application there is no scope of privaet assembely.

5. What is an Assembly ?
   Assemblies are the fundamental buildung block of .NET framework. They contains the type and resources that are useful to make an application. Assembly enables code reuse, version control, security and deployment. An assembely can have four parts : Menifest, Type metadata, MSIL and Resource file

   5. What is GAC ?
   GAC (global assembelu cache) Its an space (directory C:\winnt\assembely) on the server where all the shared assemblies are registrered and that can be used in the application for code reuse.

   6. What do you know about Machine.Config file ?
   Its a base configuration file for all .NET assemblies running on the

server. It specifies a settings that are global to a perticular machine.

7. Different types of authentication modes in .NET Framework ?
Windows, Forms, Passport and None.

8. What is Strong name ?
Strong name ensures the uniqueness of assembely on the server. A strong name incudes information about Assembely version, Public/Private Key token, Culture information and ASsembely name.

9. Where does the GAC exist ?
By defauit C:\\assembely e.g c:\winnt\assembely or c:\windows\assembely

10. What are different types that a variable can be defined and their scopes ?
Public- Can be accessed anywhere
Private- anywhere in the same class
Protected -winthin the class and the class that inherites this class
Friend- Members of the class within the assembely
Protected friend- member of assembely or inheriting class

11. What is DLL HELL ?
Previously (when using VB) we can have a situation that we have to put same name dll file in a single directory , but the dlls are of different versions. This is known as dll hell.

What is COM, COM+ and DCOM ?
COM (Component Object Model) A standard that is used to for communication between OS and the softwares. COM is used to create reusable software components
COM+ : COM+ is an extension of Component Object Model (COM). COM+ is both an OOP architecture and a set of operating system services.
DCOM an extension of the Component Object Model (COM) that allows COM components to communicate across network boundaries. Traditional COM components can only perform interprocess communication across process boundaries on the same machine. DCOM uses the RPC mechanism to transparently send and receive information between COM components (i.e., clients and servers) on the same network.

13. What is boxing and unboxing ?
Implicit (manual) conversion of value type to reference type of a variable is known as BOXING, for example integer to object type conversion. Conversion of Boxed type variable back to value type is called as UnBoxing.

14. what is connected and diconnected database ?
Connected and Disconneted database basicallythe approch that how you handle the database connection, It may be connected that once the

application starts you have to open the connection only for a single time and then performs many transactions and close the connection just before exit the application. This approch will be generally used in windows based application. On other hand disconnected architecter refere to open and close the connection for each time while performing a transactio.

15. What is garbage collection and how it works ?
Garbage Collection is Automatic Memory Manager for the dotnet framework. It manages the memory allocated to the .NET framework. CLR takes cares about .NET framework. When a variable is defined, Its gets a space in the memory and when the program control comes out of that function the scope of variable gets ended, so the garbage collection acts on and memory will releases.

When was .NET announced?
Bill Gates delivered a keynote at Forum 2000, held June 22, 2000, outlining the .NET 'vision'. The July 2000 PDC had a number of sessions on .NET technology, and delegates were given CDs containing a pre-release version of the .NET framework/SDK and Visual Studio.NET.

When was the first version of .NET released?
The final version of the 1.0 SDK and runtime was made publicly available around 6pm PST on 15-Jan-2002. At the same time, the final version of Visual Studio.NET was made available to MSDN subscribers.

What platforms does the .NET Framework run on?
The runtime supports Windows XP, Windows 2000, NT4 SP6a and Windows ME/98. Windows 95 is not supported. Some parts of the framework do not work on all platforms - for example, ASP.NET is only supported on Windows XP and Windows 2000. Windows 98/ME cannot be used for development. IIS is not supported on Windows XP Home Edition, and so cannot be used to host ASP.NET. However, the ASP.NET Web Matrix
web server does run on XP Home.
The Mono project is attempting to implement the .NET framework on Linux.

What is the CLR?
CLR = Common Language Runtime. The CLR is a set of standard resources that (in theory) any .NET program can take advantage of, regardless of programming language. Robert Schmidt (Microsoft) lists the following CLR resources in his MSDN PDC# article:
Object-oriented programming model (inheritance, polymorphism, exception handling, garbage collection)
Security model
Type system
All .NET base classes
Many .NET framework classes
Development, debugging, and profiling tools
Execution and code management

IL-to-native translators and optimizers
What this means is that in the .NET world, different programming languages will be more equal in capability than they have ever been before, although clearly not all languages will support all CLR services.

What is the CTS?
CTS = Common Type System. This is the range of types that the .NET runtime understands, and therefore that .NET applications can use. However note that not all .NET languages will support all the types in the CTS. The CTS is a superset of the CLS.

What is the CLS?
CLS = Common Language Specification. This is a subset of the CTS which all .NET languages are expected to support. The idea is that any program which uses CLS-compliant types can interoperate with any .NET program written in any language.
In theory this allows very tight interop between different .NET languages - for example allowing a C# class to inherit from a VB class.

What is IL?
IL = Intermediate Language. Also known as MSIL (Microsoft Intermediate Language) or CIL (Common Intermediate Language). All .NET source code (of any language) is compiled to IL. The IL is then converted to machine code at the point where the software is installed, or at run-time by a Just-In-Time (JIT) compiler.

What does 'managed' mean in the .NET context?
The term 'managed' is the cause of much confusion. It is used in various places within .NET, meaning slightly different things.Managed code: The .NET framework provides several core run-time services to the programs that run within it - for example
exception handling and security. For these services to work, the code must provide a minimum level of information to the runtime.
Such code is called managed code. All C# and Visual Basic.NET code is managed by default. VS7 C++ code is not managed by default, but the compiler can produce managed code by specifying a command-line switch (/com+).
Managed data: This is data that is allocated and de-allocated by the .NET runtime's garbage collector. C# and VB.NET data is always managed. VS7 C++ data is unmanaged by default, even when using the /com+ switch, but it can be marked as managed using the __gc keyword.Managed classes: This is usually referred to in the context of Managed Extensions (ME) for C++. When using ME C++, a class can be marked with the __gc keyword. As the name suggests, this means that the memory for instances of the class is managed by the garbage collector, but it also means more than that. The class becomes a fully paid-up member of the .NET community with the benefits and restrictions that brings. An example of a benefit is proper interop with classes written in other languages - for example, a managed

C++ class can inherit from a VB class. An example of a restriction is that a managed class can only inherit from one base class.

What is reflection?

All .NET compilers produce metadata about the types defined in the modules they produce. This metadata is packaged along with the module (modules in turn are packaged together in assemblies), and can be accessed by a mechanism called reflection. The System.Reflection namespace contains classes that can be used to interrogate the types for a module/assembly.
Using reflection to access .NET metadata is very similar to using ITypeLib/ITypeInfo to access type library data in COM, and it is used for similar purposes - e.g. determining data type sizes for marshaling data across context/process/machine boundaries.
Reflection can also be used to dynamically invoke methods (see System.Type.InvokeMember) , or even create types dynamically at run-time (see System.Reflection.Emit.TypeBuilder).

What is the difference between Finalize and Dispose (Garbage collection) ?

Class instances often encapsulate control over resources that are not managed by the runtime, such as window handles (HWND), database connections, and so on. Therefore, you should provide both an explicit and an implicit way to free those resources. Provide implicit control by implementing the protected Finalize Method on an object (destructor syntax in C# and the Managed Extensions for C++). The garbage collector calls this method at some point after there are no longer any valid references to the object. In some cases, you might want to provide programmers using an object with the ability to explicitly release these external resources before the garbage collector frees the object. If an external resource is scarce or expensive, better performance can be achieved if the programmer explicitly releases resources when they are no longer being used. To provide explicit control, implement the Dispose method provided by the IDisposable Interface. The consumer of the object should call this method when it is done using the object.
Dispose can be called even if other references to the object are alive. Note that even when you provide explicit control by way of Dispose, you should provide implicit cleanup using the Finalize method. Finalize provides a backup to prevent resources from
permanently leaking if the programmer fails to call Dispose.

What is Partial Assembly References?
Full Assembly reference: A full assembly reference includes the assembly's text name, version, culture, and public key token (if the assembly has a strong name). A full assembly reference is required if you reference any assembly that is part of the common
language runtime or any assembly located in the global assembly cache.

Partial Assembly reference: We can dynamically reference an assembly by providing only partial information, such as specifying only the assembly name. When you specify a partial assembly reference, the runtime looks for the assembly only in the application
directory.
We can make partial references to an assembly in your code one of the following ways:
-> Use a method such as System.Reflection.Assembly.Load and specify only a partial reference. The runtime checks for the assembly in the application directory.
-> Use the System.Reflection.Assembly.LoadWithPartialName method and specify only a partial reference. The runtime checks for the assembly in the application directory and in the global assembly cache

Changes to which portion of version number indicates an incompatible change?
Major or minor. Changes to the major or minor portion of the version number indicate an incompatible change. Under this convention then, version 2.0.0.0 would be considered incompatible with version 1.0.0.0. Examples of an incompatible change would be a change to the types of some method parameters or the removal of a type or method altogether. Build. The Build number is typically used to distinguish between daily builds or smaller compatible releases. Revision. Changes to the revision number are typically reserved for an incremental build needed to fix a particular bug. You'll sometimes hear this referred to as the "emergency bug fix" number in that the revision is what is often changed when a fix to a specific bug is shipped to a customer

What is side-by-side execution? Can two application one using private assembly and other using Shared assembly be stated as a side-by-side executables?
Side-by-side execution is the ability to run multiple versions of an application or component on the same computer. You can have multiple versions of the common language runtime, and multiple versions of applications and components that use a version of the runtime, on the same computer at the same time. Since versioning is only applied to shared assemblies, and not to private assemblies, two application one using private assembly and one using shared assembly cannot be stated as side-by-side
executables.

Why string are called Immutable data Type ?
The memory representation of string is an Array of Characters, So on re-assigning the new array of Char is formed & the start address is changed . Thus keeping the Old string in Memory for Garbage Collector to be disposed.

What does assert() method do?
In debug compilation, assert takes in a Boolean condition as a parameter, and shows the error dialog if the condition is false. The program proceeds without any interruption if the condition is true.

What's the difference between the Debug class and Trace class?
Documentation looks the same. Use Debug class for debug builds, use Trace class for both debug and release builds.

Why are there five tracing levels in System.Diagnostics.TraceSwitcher?
The tracing dumps can be quite verbose. For applications that are constantly running you run the risk of overloading the machine and the hard drive. Five levels range from None to Verbose, allowing you to fine-tune the tracing activities.

Where is the output of TextWriterTraceListener redirected?
To the Console or a text file depending on the parameter passed to the constructor.

How do assemblies find each other?
By searching directory paths. There are several factors which can affect the path (such as the AppDomain host, and application configuration files), but for private assemblies the search path is normally the application's directory and its sub-directories. For shared assemblies, the search path is normally same as the private assembly path plus the shared assembly cache.

How does assembly versioning work?
Each assembly has a version number called the compatibility version. Also each reference to an assembly (from another assembly) includes both the name and version of the referenced assembly.The version number has four numeric parts (e.g. 5.5.2.33). Assemblies with either of the first two parts different are normally viewed as incompatible. If the first two parts are the same, but the third is different, the assemblies are deemed as 'maybe compatible'. If only the fourth part is different, the assemblies are deemed compatible. However, this is just the default guideline - it is the version policy that decides to what extent these rules are enforced. The version policy can be specified via the application configuration file.

What is garbage collection?
Garbage collection is a system whereby a run-time component takes responsibility for managing the lifetime of objects and the heap memory that they occupy. This concept is not new to .NET - Java and many other languages/runtimes have used garbage collection for some time.

Why doesn't the .NET runtime offer deterministic destruction?
Because of the garbage collection algorithm. The .NET garbage collector works by periodically running through a list of all the objects that are currently being referenced by an application. All the objects that it doesn't find during this search are ready to be destroyed and the memory reclaimed. The implication of this algorithm is that the runtime doesn't get notified immediately when the final reference on an object goes away - it only finds out during the next sweep of the heap.
Futhermore, this type of algorithm works best by performing the garbage collection sweep as rarely as possible. Normally heap exhaustion is the trigger for a collection sweep.

Is the lack of deterministic destruction in .NET a problem?
It's certainly an issue that affects component design. If you have objects that maintain expensive or scarce resources (e.g. database locks), you need to provide some way for the client to tell the object to release the resource when it is done. Microsoft recommend that you provide a method called Dispose() for this purpose. However, this causes problems for distributed objects - in a distributed system who calls the Dispose() method? Some form of reference-counting or ownership-management mechanism is needed to handle distributed objects - unfortunately the runtime offers no help with this.

What is serialization?
Serialization is the process of converting an object into a stream of bytes. Deserialization is the opposite process of creating an object from a stream of bytes. Serialization / Deserialization is mostly used to transport objects (e.g. during remoting), or to persist
objects (e.g. to a file or database).

Does the .NET Framework have in-built support for serialization?
There are two separate mechanisms provided by the .NET class library - XmlSerializer and SoapFormatter/BinaryFormatter. Microsoft uses XmlSerializer for Web Services, and uses SoapFormatter/BinaryFormatter for remoting. Both are available for use in your own code.

Can I customise the serialization process?
Yes. XmlSerializer supports a range of attributes that can be used to configure serialization for a particular class. For example, a field or property can be marked with the [XmlIgnore] attribute to exclude it from serialization. Another example is the [XmlElement]

attribute, which can be used to specify the XML element name to be used for a particular property or field.
Serialization via SoapFormatter/BinaryFormatter can also be controlled to some extent by attributes. For example, the [NonSerialized] attribute is the equivalent of XmlSerializer's [XmlIgnore] attribute. Ultimate control of the serialization process can be acheived by implementing the the ISerializable interface on the class whose instances are to be serialized.

Why is XmlSerializer so slow?
There is a once-per-process-per-type overhead with XmlSerializer. So the first time you serialize or deserialize an object of a given type in an application, there is a significant delay. This normally doesn't matter, but it may mean, for example, that XmlSerializer is a poor choice for loading configuration settings during startup of a GUI application.

Why do I get errors when I try to serialize a Hashtable?
XmlSerializer will refuse to serialize instances of any class that implements IDictionary, e.g. Hashtable. SoapFormatter and BinaryFormatter do not have this restriction.

What are attributes?
There are at least two types of .NET attribute. The first type I will refer to as a metadata attribute - it allows some data to be attached to a class or method. This data becomes part of the metadata for the class, and (like other class metadata) can be accessed via reflection.
The other type of attribute is a context attribute. Context attributes use a similar syntax to metadata attributes but they are fundamentally different. Context attributes provide an interception mechanism whereby instance activation and method calls can be
pre- and/or post-processed.

How does CAS work?
The CAS security policy revolves around two key concepts - code groups and permissions. Each .NET assembly is a member of a particular code group, and each code group is granted the permissions specified in a named permission set.
For example, using the default security policy, a control downloaded from a web site belongs to the 'Zone - Internet' code group, which adheres to the permissions defined by the 'Internet' named permission set. (Naturally the 'Internet' named permission set represents a very restrictive range of permissions.)

Who defines the CAS code groups?
Microsoft defines some default ones, but you can modify these and even create your own. To see the code groups defined on your system, run 'caspol -lg' from the command-line. On my system it looks like this:
Level = Machine
Code Groups:

1. All code: Nothing
   1.1. Zone - MyComputer: FullTrust
      1.1.1. Honor SkipVerification requests: SkipVerification
   1.2. Zone - Intranet: LocalIntranet
   1.3. Zone - Internet: Internet
   1.4. Zone - Untrusted: Nothing
   1.5. Zone - Trusted: Internet
   1.6. StrongName -
00240000048000009400000006020000002400005253413100040000030
00000CFCB3291AA715FE99D40D49040336F9056D7886FED46775BC7BB54
30BA4444FEF8348EBD06
F962F39776AE4DC3B7B04A7FE6F49F25F740423EBF2C0B89698D8D08AC48
D69CED0FC8F83B465E08
07AC11EC1DCC7D054E807A43336DDE408A5393A48556123272CEEEE72F16
60B71927D38561AABF5C
AC1DF1734633C602F8F2D5:

Note the hierarchy of code groups - the top of the hierarchy is the most general ('All code'), which is then sub-divided into several groups, each of which in turn can be sub-divided. Also note that (somewhat counter-intuitively) a sub-group can be associated with a more permissive permission set than its parent.

How do I define my own code group?

Use caspol. For example, suppose you trust code from www.mydomain.com and you want it have full access to your system, but you want to keep the default restrictions for all other internet sites. To achieve this, you would add a new code group as a sub-group of the 'Zone - Internet' group, like this:

caspol -ag 1.3 -site www.mydomain.com FullTrust

Now if you run caspol -lg you will see that the new group has been added as group 1.3.1:

   1.3. Zone - Internet: Internet
      1.3.1. Site - www.mydomain.com: FullTrust

Note that the numeric label (1.3.1) is just a caspol invention to make the code groups easy to manipulate from the command-line. The underlying runtime never sees it.

How do I change the permission set for a code group?

Use caspol. If you are the machine administrator, you can operate at the 'machine' level - which means not only that the changes you make become the default for the machine, but also that users cannot change the permissions to be more permissive. If you are a normal (non-admin) user you can still modify the permissions, but only to make them more restrictive. For example, to allow intranet code to do what it likes you might do this:

caspol -cg 1.2 FullTrust

Note that because this is more permissive than the default policy (on a

standard system), you should only do this at the machine level - doing it at the user level will have no effect.

I can't be bothered with all this CAS stuff. Can I turn it off?
Yes, as long as you are an administrator. Just run: caspol -s off

Can I look at the IL for an assembly?
Yes. MS supply a tool called Ildasm which can be used to view the metadata and IL for an assembly.

Can source code be reverse-engineered from IL?
Yes, it is often relatively straightforward to regenerate high-level source (e.g. C#) from IL.

How can I stop my code being reverse-engineered from IL?
There is currently no simple way to stop code being reverse-engineered from IL. In future it is likely that IL obfuscation tools will become available, either from MS or from third parties. These tools work by 'optimising' the IL in such a way that reverse-engineering becomes much more difficult.
Of course if you are writing web services then reverse-engineering is not a problem as clients do not have access to your IL.

Is there built-in support for tracing/logging?
Yes, in the System.Diagnostics namespace. There are two main classes that deal with tracing - Debug and Trace. They both work in a similar way - the difference is that tracing from the Debug class only works in builds that have the DEBUG symbol defined, whereas tracing from the Trace class only works in builds that have the TRACE symbol defined. Typically this means that you should use System.Diagnostics.Trace.WriteLine for tracing that you want to work in debug and release builds, and System.Diagnostics.Debug.WriteLine for tracing that you want to work only in debug builds.

Can I redirect tracing to a file?
Yes. The Debug and Trace classes both have a Listeners property, which is a collection of sinks that receive the tracing that you send via Debug.WriteLine and Trace.WriteLine respectively. By default the Listeners collection contains a single sink, which is an
instance of the DefaultTraceListener class. This sends output to the Win32 OutputDebugString() function and also the
System.Diagnostics.Debugger.Log() method. This is useful when debugging, but if you're trying to trace a problem at a customer site, redirecting the output to a file is more appropriate. Fortunately, the TextWriterTraceListener class is provided for this purpose.

What are the contents of assembly?
In general, a static assembly can consist of four elements:
The assembly manifest, which contains assembly metadata.
Type metadata.

Microsoft intermediate language (MSIL) code that implements the types.
A set of resources.

What is GC (Garbage Collection) and how it works

One of the good features of the CLR is Garbage Collection, which runs in the background collecting unused object references, freeing us from having to ensure we always destroy them. In reality the time difference between you releasing the object instance and it being garbage collected is likely to be very small, since the GC is always running.
[The process of transitively tracing through all pointers to actively used objects in order to locate all objects that can be referenced, and then arranging to reuse any heap memory that was not found during this trace. The common language runtime garbage collector also compacts the memory that is in use to reduce the working space needed for the heap.]

Heap:
A portion of memory reserved for a program to use for the temporary storage of data structures whose existence or size cannot be determined until the program is running.

Differnce between Managed code and unmanaged code ?

Managed Code:
Code that runs under a "contract of cooperation" with the common language runtime. Managed code must supply the metadata necessary for the runtime to provide services such as memory management, cross-language integration, code access security, and
automatic lifetime control of objects. All code based on Microsoft intermediate language (MSIL) executes as managed code.

Un-Managed Code:
Code that is created without regard for the conventions and requirements of the common language runtime. Unmanaged code executes in the common language runtime environment with minimal services (for example, no garbage collection, limited debugging, and so on).

What is MSIL, IL, CTS and, CLR ?

MSIL: (Microsoft intermediate language)
When compiling to managed code, the compiler translates your source code into Microsoft intermediate language (MSIL), which is a CPU-independent set of instructions that can be efficiently converted to native code. MSIL includes instructions for loading, storing, initializing, and calling methods on objects, as well as instructions for arithmetic and logical operations, control flow, direct memory access, exception handling, and other operations. Before code can be executed, MSIL must be converted to CPU-specific code, usually by a just-in-time (JIT) compiler. Because the common language runtime supplies one or more JIT compilers for each computer architecture it supports, the same set of MSIL can be JIT-compiled and executed on any supported

architecture.
When a compiler produces MSIL, it also produces metadata. Metadata describes the types in your code, including the definition of each type, the signatures of each type's members, the members that your code references, and other data that the runtime uses at execution time. The MSIL and metadata are contained in a portable executable (PE) file that is based on and extends the published Microsoft PE and Common Object File Format (COFF) used historically for executable content. This file format, which accommodates

MSIL or native code as well as metadata, enables the operating system to recognize common language runtime images. The presence of metadata in the file along with the MSIL enables your code to describe itself, which means that there is no need for type libraries or Interface Definition Language (IDL). The runtime locates and extracts the metadata from the file as needed during execution.

IL: (Intermediate Language)
A language used as the output of a number of compilers and as the input to a just-in-time (JIT) compiler. The common language runtime includes a JIT compiler for converting MSIL to native code.

CTS: (Common Type System)
The specification that determines how the common language runtime defines, uses, and manages types

CLR: (Common Language Runtime)
The engine at the core of managed code execution. The runtime supplies managed code with services such as cross-language integration, code access security, object lifetime management, and debugging and profiling support.

What is Reference type and value type ?
Reference Type:
Reference types are allocated on the managed CLR heap, just like object types.
A data type that is stored as a reference to the value's location. The value of a reference type is the location of the sequence of bits that represent the type's data. Reference types can be self-describing types, pointer types, or interface types

Value Type:
Value types are allocated on the stack just like primitive types in VBScript, VB6 and C/C++. Value types are not instantiated using new go out of scope when the function they are defined within returns.
Value types in the CLR are defined as types that derive from system.valueType.

A data type that fully describes a value by specifying the sequence of bits that constitutes the value's representation. Type information for a value type instance is not stored with the instance at run time, but it is available in metadata. Value type instances can be treated as objects using boxing.

What is Boxing and unboxing ?
Boxing:
The conversion of a value type instance to an object, which implies that the instance will carry full type information at run time and will be allocated in the heap. The Microsoft intermediate language (MSIL) instruction set's box instruction converts a value type to an object by making a copy of the value type and embedding it in a newly allocated object.

Un-Boxing:
The conversion of an object instance to a value type.

What is JIT and how is works ?
An acronym for "just-in-time," a phrase that describes an action that is taken only when it becomes necessary, such as just-in-time compilation or just-in-time object activation

What is portable executable (PE) ?
The file format used for executable programs and for files to be linked together to form executable programs

What is strong name?
A name that consists of an assembly's identity—its simple text name, version number, and culture information (if provided)—strengthened by a public key and a digital signature generated over the assembly. Because the assembly manifest
contains file hashes for all the files that constitute the assembly implementation, it is sufficient to generate the digital signature over just the one file in the assembly that contains the assembly manifest. Assemblies with the same strong name are expected to be identical

What is global assembly cache?
A machine-wide code cache that stores assemblies specifically installed to be shared by many applications on the computer. Applications deployed in the global assembly cache must have a strong name.

What is difference between constants, readonly and, static ?
Constants: The value can’t be changed
Read-only: The value will be initialized only once from the constructor of the class.
Static: Value can be initialized once.

What is difference between shared and public?
An assembly that can be referenced by more than one application. An assembly must be explicitly built to be shared by giving it a cryptographically strong name.

What is namespace used for loading assemblies at run time and name the methods?
System.Reflection

What are the types of authentication in .net?
We have three types of authentication:
1. Form authentication
2. Windows authentication
3. Passport
This has to be declared in web.config file.

What is the difference between a Struct and a Class ?
The struct type is suitable for representing lightweight objects such as Point, Rectangle, and Color. Although it is possible to represent a point as a class, a struct is more efficient in some scenarios. For example, if you declare an array of 1000 Point objects,
you will allocate additional memory for referencing each object. In this case, the struct is less expensive.
When you create a struct object using the new operator, it gets created and the appropriate constructor is called. Unlike classes, structs can be instantiated without using the new operator. If you do not use new, the fields will remain unassigned and the object cannot be used until all of the fields are initialized. It is an error to declare a default (parameterless) constructor for a struct. A default constructor is always provided to initialize the struct members to their default values.
It is an error to initialize an instance field in a struct.
There is no inheritance for structs as there is for classes. A struct cannot inherit from another struct or class, and it cannot be the base of a class. Structs, however, inherit from the base class Object. A struct can implement interfaces, and it does that exactly as classes do.
A struct is a value type, while a class is a reference type.
How big is the datatype int in .NET?
32 bits.

How big is the char?
16 bits (Unicode).

How do you initiate a string without escaping each backslash?
Put an @ sign in front of the double-quoted string.

What's the access level of the visibility type internal?
Current application.

Explain encapsulation ?
The implementation is hidden, the interface is exposed.

What data type should you use if you want an 8-bit value that's signed?
sbyte.

Speaking of Boolean data types, what's different between C# and C/C++?
There's no conversion between 0 and false, as well as any other number and true, like in C/C++.

Where are the value-type variables allocated in the computer RAM?
Stack.

Where do the reference-type variables go in the RAM?
The references go on the stack, while the objects themselves go on the heap.

What is the difference between the value-type variables and reference-type variables in terms of garbage collection?
The value-type variables are not garbage-collected, they just fall off the stack when they fall out of scope, the reference-type objects are picked up by GC when their references go null.

How do you convert a string into an integer in .NET?
Int32.Parse(string)

How do you box a primitive data type variable?
Assign it to the object, pass an object.

Why do you need to box a primitive variable?
To pass it by reference.

What's the difference between Java and .NET garbage collectors?
Sun left the implementation of a specific garbage collector up to the JRE developer, so their performance varies widely, depending on whose JRE you're using. Microsoft standardized on their garbage collection.

How do you enforce garbage collection in .NET?
System.GC.Collect();

What's different about namespace declaration when comparing that to package declaration in Java?
No semicolon.

What's the difference between const and readonly?
You can initialize readonly variables to some runtime values. Let's say your program uses current date and time as one of the values that won't change. This way you declare public readonly string DateT = new DateTime().ToString().

What happens when you encounter a continue statement inside the for loop?
The code for the rest of the loop is ignored, the control is transferred back to the beginning of the loop.

What's the advantage of using System.Text.StringBuilder over System.String?
StringBuilder is more efficient in the cases, where a lot of manipulation is done to the text. Strings are immutable, so each time it's being operated on, a new instance is created.

Can you store multiple data types in System.Array?
No.

What's the difference between the System.Array.CopyTo() and System.Array.Clone()?
The first one performs a deep copy of the array, the second one is shallow.

How can you sort the elements of the array in descending order?
By calling Sort() and then Reverse() methods.

What's the .NET datatype that allows the retrieval of data by a unique key?
HashTable.

What's class SortedList underneath?
A sorted HashTable.

Will finally block get executed if the exception had not occurred?
Yes.

Can multiple catch blocks be executed?
No, once the proper catch code fires off, the control is transferred to the finally block (if there are any), and then whatever follows the finally block.

Why is it a bad idea to throw your own exceptions?
Well, if at that point you know that an error has occurred, then why not write the proper code to handle that error instead of passing a new Exception object to the catch block? Throwing your own exceptions signifies some design flaws in the project.

What's a delegate?
A delegate object encapsulates a reference to a method. In C++ they were referred to as function pointers.

What's a multicast delegate?
It's a delegate that points to and eventually fires off several methods.

How's the DLL Hell problem solved in .NET?
Assembly versioning allows the application to specify not only the library it needs to run (which was available under Win32), but also the version of the assembly.

What are the ways to deploy an assembly?
An MSI installer, a CAB archive, and XCOPY command.

What's a satellite assembly?
When you write a multilingual or multi-cultural application in .NET, and want to distribute the core application separately from the localized modules, the localized assemblies that modify the core application are called satellite assemblies.

What namespaces are necessary to create a localized application?
System.Globalization, System.Resources

What does assert() do?
In debug compilation, assert takes in a Boolean condition as a parameter, and shows the error dialog if the condition is false. The program proceeds without any interruption if the condition is true.

What's the difference between the Debug class and Trace class?
Documentation looks the same. Use Debug class for debug builds, use Trace class for both debug and release builds.

Why are there five tracing levels in System.Diagnostics.TraceSwitcher?
The tracing dumps can be quite verbose and for some applications that are constantly running you run the risk of overloading the machine and the hard drive there. Five levels range from None to Verbose, allowing to fine-tune the tracing activities.

Where is the output of TextWriterTraceListener redirected?
To the Console or a text file depending on the parameter passed to the constructor.

What namespaces are necessary to create a localized application?
System.Globalization, System.Resources.

What are three test cases you should go through in unit testing?
Positive test cases (correct data, correct output), negative test cases (broken or missing data, proper handling), exception test cases (exceptions are thrown and caught properly).

Can you change the value of a variable while debugging a C# application?
Yes, if you are debugging via Visual Studio.NET, just go to Immediate window.

What's the implicit name of the parameter that gets passed into the class' set method?
Value, and it's datatype depends on whatever variable we're changing.

How do you inherit from a class in C#?
Place a colon and then the name of the base class. Notice that it's double colon in C++.

Does C# support multiple inheritance?
No, use interfaces instead.

When you inherit a protected class-level variable, who is it available to?
Derived Classes.

What's the top .NET class that everything is derived from?
System.Object.

How's method overriding different from overloading?
When overriding, you change the method behavior for a derived class. Overloading simply involves having a method with the same name within the class.

What does the keyword virtual mean in the method definition?
The method can be over-ridden.

Can you declare the override method static while the original method is non-static?
No, you can't, the signature of the virtual method must remain the same, only the keyword virtual is changed to keyword override.

Can you override private virtual methods?
No, moreover, you cannot access private methods in inherited classes, have to be protected in the base class to allow any sort of access.

Can you prevent your class from being inherited and becoming a base class for some other classes?
Yes, that's what keyword sealed in the class definition is for. The developer trying to derive from your class will get a message: cannot inherit from Sealed class WhateverBaseClassName. It's the same concept as final class in Java.

Can you allow class to be inherited, but prevent the method from being over-ridden?
Yes, just leave the class public and make the method sealed.

Why can't you specify the accessibility modifier for methods inside the interface?
They all must be public. Therefore, to prevent you from getting the false impression that you have any freedom of choice, you are not allowed to specify any accessibility, it's public by default.

Can you inherit multiple interfaces?
Yes, why not.

And if they have conflicting method names?
It's up to you to implement the method inside your own class, so implementation is left entirely up to you. This might cause a problem on a higher-level scale if similarly named methods from different interfaces expect different data, but as far as compiler cares you're
okay.

What's the difference between an interface and abstract class?
In the interface all methods must be abstract, in the abstract class some methods can be concrete. In the interface no accessibility modifiers are allowed, which is ok in abstract classes.

How can you overload a method?
Different parameter data types, different number of parameters, different order of parameters.

If a base class has a bunch of overloaded constructors, and an inherited class has another bunch of overloaded constructors, can you enforce a call from an inherited constructor to an arbitrary base constructor?
Yes, just place a colon, and then keyword base (parameter list to invoke the appropriate constructor) in the overloaded constructor definition inside the inherited class.

What's the difference between System.String and System.StringBuilder classes?
System.String is immutable, System.StringBuilder was designed with the purpose of having a mutable string where a variety of operations can be performed.

Does C# support multiple-inheritance?
No, use interfaces instead.

When you inherit a protected class-level variable, who is it available to?
The derived class.

Are private class-level variables inherited?
Yes, but they are not accessible. Although they are not visible or accessible via the class interface, they are inherited.

Describe the accessibility modifier "protected internal".
It is available to derived classes and classes within the same Assembly (and naturally from the base class it's declared in).

What's the top .NET class that everything is derived from?
System.Object.

What's the advantage of using System.Text.StringBuilder over System.String?
StringBuilder is more efficient in cases where there is a large amount of string manipulation. Strings are immutable, so each time it's being operated on, a new instance is created.

Can you store multiple data types in System.Array?
No.

What's the .NET class that allows the retrieval of a data element using a unique key?
HashTable.

Will the finally block get executed if an exception has not occurred?
Yes.

What's an abstract class?
A class that cannot be instantiated. An abstract class is a class that must be inherited and have the methods overridden. An abstract class is essentially a blueprint for a class without any implementation.

When do you absolutely have to declare a class as abstract?

1. When at least one of the methods in the class is abstract.
2. When the class itself is inherited from an abstract class, but not all base abstract methods have been overridden.

What's an interface?

It's an abstract class with public abstract methods all of which must be implemented in the inherited classes.

Why can't you specify the accessibility modifier for methods inside the interface?

They all must be public. Therefore, to prevent you from getting the false impression that you have any freedom of choice,
you are not allowed to specify any accessibility, it's public by default.

What's the difference between an interface and abstract class?

In an interface class, all methods must be abstract. In an abstract class some methods can be concrete. In an interface class, no accessibility modifiers are allowed, which is ok in an abstract class.

How is method overriding different from method overloading?

When overriding a method, you change the behavior of the method for the derived class. Overloading a method simply involves
having another method with the same name within the class.

Can you declare an override method to be static if the original method is non-static?

No. The signature of the virtual method must remain the same, only the keyword virtual is changed to keyword override.

Can you override private virtual methods?

No. Private methods are not accessible outside the class.

Can you write a class without specifying namespace? Which namespace does it belong to by default?

Yes, you can, then the class belongs to global namespace which has no name. For commercial products, naturally, you wouldn't want global namespace

What is a formatter?

A formatter is an object that is responsible for encoding and serializing data into messages on one end, and deserializing and decoding messages into data on the other end.

Different b/w .NET & J2EE ?
Differences between J2EE and the .NET Platform

Vendor Neutrality

The .NET platform is not vendor neutral, it is tied to the Microsoft operating systems. But neither are any of the J2EE implementations
Many companies buy into J2EE believing that it will give them vendor neutrality. And, in fact, this is a stated goal of Sun's vision:
A wide variety of J2EE product configurations and implementations, all of which meet the requirements of this specification, are possible. A portable J2EE application will function correctly when successfully deployed in any of these products. (ref : Java 2 Platform Enterprise Edition Specification, v1.3, page 2-7 available at http://java.sun.com/j2ee/)

Overall Maturity

Given that the .NET platform has a three year lead over J2EE, it should be no surprise to learn that the .NET platform is far more mature than the J2EE platform. Whereas we have high volume highly reliable web sites using .NET technologies (NASDAQ and Dell being among many examples)

Interoperability and Web Services

The .NET platform eCollaboration model is, as I have discussed at length, based on the UDDI and SOAP standards. These standards are widely supported by more than 100 companies. Microsoft, along with IBM and Ariba, are the leaders in this area. Sun is a member of the UDDI consortium and recognizes the importance of the UDDI standards. In a recent press release, Sun's George Paolini, Vice President for the Java Community Development, says:

"Sun has always worked to help establish and support open, standards-based technologies that facilitate the growth of network-based applications, and we see UDDI as an important project to establish a registry framework for business-to-business e-commerce

But while Sun publicly says it believes in the UDDI standards, in reality, Sun has done nothing whatsoever to incorporate any of the UDDI standards into J2EE.

Scalability

Typical Comparision w.r.t Systems and their costs

J2EE

Company	System	Total Sys.	Cost
Bull	Escala T610 c/s	16,785	$1,980,179
IBM	RS/6000 Enterprise Server F80	16,785	$2,026,681
Bull	Escala EPC810 c/s	33,375	$3,037,499
IBM	RS/6000 Enterprise Server M80	33,375	$3,097,055
Bull	Escala EPC2450	110,403	$9,563,263
IBM	IBM eServer pSeries 680 Model 7017-S85	110,403	$9,560,594

.NET platform systems

Company	System	Total Sys.	Cost
Dell	PowerEdge 4400	16,263	$273,487
Compaq	ProLiant ML-570-6/700-3P	20,207	$201,717
Dell	PowerEdge 6400	30,231	$334,626
IBM	Netfinity 7600 c/s	32,377	$443,463
Compaq	ProLiant 8500-X550-64P	161,720	$3,534,272
Compaq	ProLiant 8500-X700-64P	179,658	$3,546,582
Compaq	ProLiant 8500-X550-96P	229,914	$5,305,571
Compaq	ProLiant 8500-X700-96P	262,244	$5,305,571
Compaq	ProLiant 8500-700-192P	505,303	$10,003,826

Framework Support

The .NET platform includes such an eCommerce framework called Commerce Server. At this point, there is no equivalent vendor-neutral framework in the J2EE space. With J2EE, you should assume that you will be building your new eCommerce solution from scratch

Moreover, no matter what [J2EE] vendor you choose, if you expect a component framework that will allow you to quickly field complete e-business applications, you are in for a frustrating experience

Language

In the language arena, the choice is about as simple as it gets. J2EE supports Java, and only Java. It will not support any other language in the foreseeable future. The .NET platform supports every language except Java (although it does support a language that is syntactically and functionally equivalent to Java, C#). In fact, given the importance of the .NET platform as a language independent vehicle, it is likely that any language that comes out in the near future will include support for the .NET platform.

Some companies are under the impression that J2EE supports other languages. Although both IBM's WebSphere and BEA's WebLogic support other languages, neither does it through their J2EE technology. There are only two official ways in the J2EE platform to access other languages, one through the Java Native Interface and the other through CORBA interoperability. Sun recommends the later approach. As Sun's Distinguished Scientist and Java Architect Rick Cattell said in a recent interview.

Portability

The reason that operating system portability is a possibility with J2EE is not so much because of any inherent portability of J2EE, as it is that most of the J2EE vendors support multiple operating systems. Therefore as long as one sticks with a given J2EE vendor and a given database vendor, moving from one operating system to another should be possible. This is probably the single most important benefit in favor of J2EE over the .NET platform, which is limited to the Windows operating system. It is worth noting, however, that Microsoft has submitted the specifications for C# and a subset of the .NET Framework (called the common language infrastructure) to ECMA, the group that standardizes JavaScript.

J2EE offers an acceptable solution to ISVs when the product must be marketed to non-Windows customers, particularly when the J2EE platform itself can be bundled with the ISV's product as an integrated offering.

If the primary customer base for the ISV is Windows customers, then the .NET platform should be chosen. It will provide much better performance at a much lower cost.

Client device independence

The major difference being that with Java, it is the presentation tier programmer that determines the ultimate HTML that will be delivered to the client, and with .NET, it is a Visual Studio.NET control.

This Java approach has three problems. First, it requires a lot of code on the presentation tier, since every possible thin client system requires a different code path. Second, it is very difficult to test the code with every possible thin client system. Third, it is very difficult to add new thin clients to an existing application, since to do so involves searching through, and modifying a tremendous amount of presentation tier logic.

The .NET Framework approach is to write device independent code that interacts with visual controls. It is the control, not the programmer, that is responsible for determining what HTML to deliver, based on the capabilities of the client device.. In the .NET Framework model, one can forget that such a thing as HTML even exists!

Sun's J2EE vision is based on a family of specifications that can be implemented by many vendors. It is open in the sense that any company can license and implement the technology, but closed in the sense that it is controlled by a single vendor, and a self contained architectural island with very limited ability to interact outside of itself. One of J2EE's major disadvantages is that the choice of the platform dictates the use of a single programming language, and a programming language that is not well suited for most businesses. One of J2EE's major advantages is that most of the J2EE vendors do offer operating system portability.

Microsoft's .NET platform vision is a family of products rather than specifications, with specifications used primarily to define points of interoperability. The major disadvantage of this approach is that if is limited to the Windows platform, so applications written for the .NET platform can only be run on .NET platforms. Their are several important advantages to the .NET platform:

* The cost of developing applications is much lower, since standard business languages can be used and device independent presentation tier logic can be written.

* The cost of running applications is much lower, since commodity hardware platforms (at 1/5 the cost of their Unix counterparts) can be used.

* The ability to scale up is much greater, with the proved ability to support at least ten times the number of clients any J2EE platform has shown itself able to support.

* Interoperability is much stronger, with industry standard eCollaboration built into the platform.

What are the Main Features of .NET platform?

Features of .NET Platform are :-

Common Language Runtime
Explains the features and benefits of the common language runtime, a run-time environment that manages the execution of code and provides services that simplify the development process.

Assemblies
Defines the concept of assemblies, which are collections of types and resources that form logical units of functionality. Assemblies are the fundamental units of deployment, version control, reuse, activation scoping, and security permissions.

Application Domains
Explains how to use application domains to provide isolation between applications.

Runtime Hosts
Describes the runtime hosts supported by the .NET Framework, including ASP.NET, Internet Explorer, and shell executables.

Common Type System
Identifies the types supported by the common language runtime.

Metadata and Self-Describing Components
Explains how the .NET Framework simplifies component interoperation by allowing compilers to emit additional declarative information, or metadata, into all modules and assemblies.

Cross-Language Interoperability
Explains how managed objects created in different programming languages can interact with one another.

.NET Framework Security
Describes mechanisms for protecting resources and code from unauthorized code and unauthorized users.

.NET Framework Class Library
Introduces the library of types provided by the .NET Framework, which expedites and optimizes the development process and gives you access to system functionality.

What is the use of JIT ?
JIT (Just - In - Time) is a compiler which converts MSIL code to Native Code (ie.. CPU-specific code that runs on the same computer architecture).

Because the common language runtime supplies a JIT compiler for each supported CPU architecture, developers can write a set of MSIL that can be JIT-compiled and run on computers with different architectures. However, your managed code will run only on a specific operating system if it calls platform-specific native APIs, or a platform-specific class library.

JIT compilation takes into account the fact that some code might never get called during execution. Rather than using time and memory to convert all the MSIL in a portable executable (PE) file to native code, it converts the MSIL as needed during execution and stores the resulting native code so that it is accessible for subsequent calls. The loader creates and attaches a stub to each of a type's methods when the type is loaded. On the initial call to the method, the stub passes control to the JIT compiler, which converts the MSIL for that method into native code and modifies the stub to direct execution to the location of the native code. Subsequent calls of the JIT-compiled method proceed directly to the native code that was previously generated, reducing the time it takes to JIT-compile and run the code.

What meant of assembly & global assembly cache (gac) & Meta data.
Assembly :-- An assembly is the primary building block of a .NET based application. It is a collection of functionality that is built, versioned, and deployed as a single implementation unit (as one or more files). All managed types and resources are marked either as accessible only within their implementation unit, or as accessible by code outside that unit. It overcomes the problem of 'dll Hell'.The .NET Framework uses assemblies as the fundamental unit for several purposes:

- Security
- Type Identity
- Reference Scope
- Versioning
- Deployment

Global Assembly Cache :-- Assemblies can be shared among multiple applications on the machine by registering them in global Assembly cache(GAC). GAC is a machine wide a local cache of assemblies maintained by the .NET Framework. We can register the assembly to global assembly cache by using gacutil command.
We can Navigate to the GAC directory, C:\winnt\Assembly in explore. In the tools menu select the cache properties; in the windows displayed you can set the memory limit in MB used by the GAC
MetaData :--Assemblies have Manifests. This Manifest contains Metadata information of the Module/Assembly as well as it contains detailed Metadata

of other assemblies/modules references (exported). It's the Assembly Manifest which differentiates between an Assembly and a Module.

What are the mobile devices supported by .net platform
The Microsoft .NET Compact Framework is designed to run on mobile devices such as mobile phones, Personal Digital Assistants (PDAs), and embedded devices. The easiest way to develop and test a Smart Device Application is to use an emulator.

These devices are divided into two main divisions:
1) Those that are directly supported by .NET (Pocket PCs, i-Mode phones, and WAP devices)
2) Those that are not (Palm OS and J2ME-powered devices).

What is GUID , why we use it and where?
GUID :-- GUID is Short form of Globally Unique Identifier, a unique 128-bit number that is produced by the Windows OS or by some Windows applications to identify a particular component, application, file, database entry, and/or user. For instance, a Web site may generate a GUID and assign it to a user's browser to record and track the session. A GUID is also used in a Windows registry to identify COM DLLs. Knowing where to look in the registry and having the correct GUID yields a lot information about a COM object (i.e., information in the type library, its physical location, etc.). Windows also identifies user accounts by a username (computer/domain and username) and assigns it a GUID. Some database administrators even will use GUIDs as primary key values in databases.

GUIDs can be created in a number of ways, but usually they are a combination of a few unique settings based on specific point in time (e.g., an IP address, network MAC address, clock date/time, etc.).

Describe the difference between inline and code behind - which is best in a loosely coupled solution
ASP.NET supports two modes of page development: Page logic code that is written inside runat="server"> blocks within an .aspx file and dynamically compiled the first time the page is requested on the server. Page logic code that is written within an external class that is compiled prior to deployment on a server and linked ""behind"" the .aspx file at run time.

Whats MSIL, and why should my developers need an appreciation of it if at all?
When compiling the source code to managed code, the compiler translates the source into Microsoft intermediate language (MSIL). This is a CPU-independent set of instructions that can efficiently be converted to native code. Microsoft intermediate language (MSIL) is a translation used as the output of a number of compilers. It is the input to a just-in-time (JIT) compiler. The Common Language Runtime includes a JIT compiler for the conversion of MSIL to native code.

Before Microsoft Intermediate Language (MSIL) can be executed it, must be converted by the .NET Framework just-in-time (JIT) compiler to native code. This is CPU-specific code that runs on the same computer architecture as the JIT compiler. Rather than using time and memory to convert all of the MSIL in a portable executable (PE) file to native code. It converts the MSIL as needed whilst executing, then caches the resulting native code so its accessible for any subsequent calls

How many .NET languages can a single .NET DLL contain?
One

What type of code (server or client) is found in a Code-Behind class?
Server

Whats an assembly?
Assemblies are the building blocks of .NET Framework applications; they form the fundamental unit of deployment, version control, reuse, activation scoping, and security permissions. An assembly is a collection of types and resources that are built to work together and form a logical unit of functionality. An assembly provides the common language runtime with the information it needs to be aware of type implementations. To the runtime, a type does not exist outside the context of an assembly.

How many classes can a single .NET DLL contain?
Unlimited.

What is the difference between string and String ?
No difference

What is manifest?
It is the metadata that describes the assemblies.

What is metadata?
Metadata is machine-readable information about a resource, or ""data about data."" Such information might include details on content, format, size, or other characteristics of a data
source. In .NET, metadata includes type definitions, version information, external assembly references, and other standardized information.

What are the types of assemblies?
There are four types of assemblies in .NET:

Static assemblies
These are the .NET PE files that you create at compile time.

Dynamic assemblies
These are PE-formatted, in-memory assemblies that you dynamically create at runtime using the classes in the System.Reflection.Emit namespace.

Private assemblies
These are static assemblies used by a specific application.

Public or shared assemblies
These are static assemblies that must have a unique shared name and can be used by any application.

An application uses a private assembly by referring to the assembly using a static path or through an XML-based application configuration file. While the CLR doesn't enforce versioning policies-checking whether the correct version is used-for private assemblies, it ensures that an
application uses the correct shared assemblies with which the application was built. Thus, an application uses a specific shared assembly by referring to the specific shared assembly, and the CLR ensures that the correct version is loaded at runtime.

In .NET, an assembly is the smallest unit to which you can associate a version number;

What are delegates?where are they used ?
A delegate defines a reference type that can be used to encapsulate a method with a specific signature. A delegate instance encapsulates a static or an instance method. Delegates are roughly similar to function pointers in C++; however, delegates are type-safe and secure.

When do you use virutal keyword?.
When we need to override a method of the base class in the sub class, then we give the virtual keyword in the base class method. This makes the method in the base class to be overridable. Methods, properties, and indexers can be virtual, which means that their implementation can be overridden in derived classes.

What are class access modifiers ?
Access modifiers are keywords used to specify the declared accessibility of a member or a type. This section introduces the four access modifiers:
· Public - Access is not restricted.
· Protected - Access is limited to the containing class or types derived from the containing class.
· Internal - Access is limited to the current assembly.
· Protected inertnal - Access is limited to the current assembly or types derived · from the containing class.
· Private - Access is limited to the containing type.

What Is Boxing And Unboxing?
Boxing :- Boxing is an implicit conversion of a value type to the type object type

Eg:-
Consider the following declaration of a value-type variable:
int i = 123;
object o = (object) i;
Boxing Conversion

UnBoxing :- Unboxing is an explicit conversion from the type object to a value type
Eg:
```
int i = 123;          // A value type
object box = i;       // Boxing
int j = (int)box;     // Unboxing
```

What is Value type and refernce type in .Net?.
Value Type : A variable of a value type always contains a value of that type. The assignment to a variable of a value type creates a copy of the assigned value, while the assignment to a variable of a reference type creates a copy of the reference but not of the referenced object.

The value types consist of two main categories:
* Stuct Type
* Enumeration Type

Reference Type :Variables of reference types, referred to as objects, store references to the actual data. This section introduces the following keywords used to declare reference types:
* Class
* Interface
* Delegate

This section also introduces the following built-in reference types:
* object
* string

What is the difference between structures and enumeration?.
Unlike classes, structs are value types and do not require heap allocation. A variable of a struct type directly contains the data of the struct, whereas a variable of a class type contains a reference to the data. They are derived from System.ValueType class.

Enum->An enum type is a distinct type that declares a set of named constants.They are strongly typed constants. They are unique types that allow to declare symbolic names to integral values. Enums are value types, which means they contain their own value, can't inherit or be inherited from and assignment copies the value of one enum to another.

```
public enum Grade
{
  A,
  B,
  C
}
```

What is namespaces?.
Namespace is a logical naming scheme for group related types.Some class types that logically belong together they can be put into a common namespace. They prevent namespace collisions and they provide scoping. They are imported as "using" in C# or "Imports" in Visual Basic. It seems as if these directives specify a particular assembly, but they don't. A namespace can span multiple assemblies, and an assembly can define multiple namespaces. When the compiler needs the definition for a class type, it tracks through each of the different imported namespaces to the type name and searches each referenced assembly until it is found.
Namespaces can be nested. This is very similar to packages in Java as far as scoping is concerned.

How do you create shared assemblies?.
Just look through the definition of Assemblies..
* An Assembly is a logical unit of code
* Assembly physically exist as DLLs or EXEs
* One assembly can contain one or more files
* The constituent files can include any file types like image files, text files etc. along with DLLs or EXEs
* When you compile your source code by default the exe/dll generated is actually an assembly
* Unless your code is bundled as assembly it can not be used in any other application
* When you talk about version of a component you are actually talking about version of the assembly to which the component belongs.
* Every assembly file contains information about itself. This information is called as Assembly Manifest.

Following steps are involved in creating shared assemblies :

* Create your DLL/EXE source code
* Generate unique assembly name using SN utility
* Sign your DLL/EXE with the private key by modifying AssemblyInfo file
* Compile your DLL/EXE
* Place the resultant DLL/EXE in global assembly cache using AL utility

What is global assembly cache?

Each computer where the common language runtime is installed has a machine-wide code cache called the global assembly cache. The global assembly cache stores assemblies specifically designated to be shared by several applications on the computer.

There are several ways to deploy an assembly into the global assembly cache:

· Use an installer designed to work with the global assembly cache. This is the preferred option for installing assemblies into the global assembly cache.

· Use a developer tool called the Global Assembly Cache tool (Gacutil.exe), provided by the .NET Framework SDK.

· Use Windows Explorer to drag assemblies into the cache.

What is MSIL?.

When compiling to managed code, the compiler translates your source code into Microsoft intermediate language (MSIL), which is a CPU-independent set of instructions that can be efficiently converted to native code. MSIL includes instructions for loading, storing, initializing, and calling methods on objects, as well as instructions for arithmetic and logical operations, control flow, direct memory access, exception handling, and other operations. Before code can be run, MSIL must be converted to CPU-specific code, usually by a just-in-time (JIT) compiler. Because the common language runtime supplies one or more JIT compilers for each computer architecture it supports, the same set of MSIL can be JIT-compiled and run on any supported architecture.

When a compiler produces MSIL, it also produces metadata. Metadata describes the types in your code, including the definition of each type, the signatures of each type's members, the members that your code references, and other data that the runtime uses at execution time. The MSIL and metadata are contained in a portable executable (PE) file that is based on and extends the published Microsoft PE and common object file format (COFF) used historically for executable content. This file format, which accommodates MSIL or native code as well as metadata, enables the operating system to recognize common language runtime images. The presence of metadata in the file along with the MSIL enables your code to describe itself, which means that there is no need for type libraries or Interface Definition Language (IDL). The runtime locates and extracts the metadata from the file as needed during execution.

What is Jit compilers?.how many are available in clr?
Just-In-Time compiler- it converts the language that you write in .Net into machine language that a computer can understand. there are tqo types of JITs one is memory optimized & other is performace optimized.

What is tracing?Where it used.Explain few methods available
Tracing refers to collecting information about the application while it is running. You use tracing information to troubleshoot an application. Tracing allows us to observe and correct programming errors. Tracing enables you to record information in various log files about the errors that might occur at run time. You can analyze these log files to find the cause of the errors.

In .NET we have objects called Trace Listeners. A listener is an object that receives the trace output and outputs it somewhere; that somewhere could be a window in your development environment, a file on your hard drive, a Windows Event log, a SQL Server or Oracle database, or any other customized data store.

The System.Diagnostics namespace provides the interfaces, classes, enumerations and structures that are used for tracing The System.Diagnostics namespace provides two classes named Trace and Debug that are used for writing errors and application execution information in logs.

All Trace Listeners have the following functions. Functionality of these functions is same except that the target media for the tracing output is determined by the Trace Listener.

Method Name
Result Fail Outputs the specified text with the Call Stack.
Write Outputs the specified text.
WriteLine Outputs the specified text and a carriage return.
Flush Flushes the output buffer to the target media.
Close Closes the output stream in order to not receive the tracing/debugging output

How to set the debug mode?
Debug Mode for ASP.NET applications - To set ASP.NET appplication in debugging mode, edit the application's web.config and assign the "debug" attribute in < compilation > section to "true" as show below:

```
< configuration >
  < system.web >
    < compilation defaultLanguage="vb" debug="true" / >
....
...
..
< / configuration >
```

This case-sensitive attribute 'debug tells ASP.NET to generate symbols for dynamically generated files and enables the
debugger to attach to the ASP.NET application. ASP.NET will detect this change automatically, without the need to restart the server. Debug Mode for ASP.NET Webservices - Debugging an XML Web service created with ASP.NET is similar to the debugging an ASP.NET Web application.

What is the property available to check if the page posted or not?
The Page_Load event handler in the page checks for IsPostBack property value, to ascertain whether the page is posted. The Page.IsPostBack gets a value indicating whether the page is being loaded in response to the client postback, or it is for the first time. The value of Page.IsPostBack is True, if the page is being loaded in response to the client postback; while its value is False, when the page is loaded for the first time. The Page.IsPostBack property facilitates execution of certain routine in Page_Load, only once (for e.g. in Page load, we need to set default value in controls, when page is loaded for the first time. On post back, we check for true value for IsPostback value and then invoke server-side code to
update data).

Which are the abstract classes available under system.xml namespace?
The System.XML namespace provides XML related processing ability in .NET framework. XmlReader and XMLWriter are the two abstract classes at the core of .NET Framework XML classes:

1. XmlReader provides a fast, forward-only, read-only cursor for processing an XML document stream.
2. XmlWriter provides an interface for producing XML document streams that conform to the W3C's XML standards.

Both XmlReader and XmlWriter are abstract base classes, which define the functionality that all derived classes must support.

Is it possible to use multipe inheritance in .net?
Multiple Inheritance is an ability to inherit from more than one base class i.e. ability of a class to have more than one superclass, by inheriting from different sources and thus combine separately-defined behaviors in a single class. There are two types of multiple inheritance: multiple type/interface inheritance and multiple implementation inheritance. C# & VB.NET supports only multiple type/interface inheritance, i.e.
you can derive an class/interface from multiple interfaces. There is no support for multiple implementation inheritance in .NET. That means a class can only derived from one class.

What are the derived classes from xmlReader and xmlWriter?
Both XmlReader and XmlWriter are abstract base classes, which define the functionality that all derived classes must support.
There are three concrete implementations of XmlReader:

1.XmlTextReader
2.XmlNodeReader
3.XmlValidatingReader

There are two concrete implementations of XmlWriter:

1.XmlTextWriter
2.XmlNodeWriter

XmlTextReader and XmlTextWriter support reading data to/from text-based stream, while XmlNodeReader and XmlNodeWriter are designed for working with in-memory DOM tree structure. The custom readers and writers can also be developed to extend the built-in functionality of XmlReader and XmlWriter.

What is managed and unmanaged code?
The .NET framework provides several core run-time services to the programs that run within it - for example exception handling and security. For these services to work, the code must provide a minimum level of information to the runtime. i.e., code executing under the control of the CLR is called managed code. For example, any code written in C# or Visual Basic .NET is managed code.

Code that runs outside the CLR is referred to as "unmanaged code." COM components, ActiveX components, and Win32 API functions are examples of unmanaged code.

How you deploy .NET assemblies?
One way is simply use xcopy. others are use and the setup projects in .net. and one more way is use of nontuch deployment.

What is Globalizationa and Localization ?
Globalization is the process of creating an application that meets the needs of users from multiple cultures. It includes using the correct
currency, date and time format, calendar, writing direction, sorting rules, and other issues. Accommodating these cultural differences in an application is called localization.Using classes of System.Globalization namespace, you can set application's current culture.

This can be achieved by using any of the following 3 approaches.

1. Detect and redirect
2. Run-time adjustment
3. Using Satellite assemblies.

Whate are Resource Files ? How are they used in .NET?
Resource files are the files containing data that is logically deployed with an application.These files can contain data in a number of formats including strings, images and persisted objects. It has the main advantage of If we store data in these files then we don't need to compile these if the data get changed. In .NET we basically require them storing culture specific informations by localizing application's resources. You can deploy your resources using satellite assemblies.

Difference between Dispose and Finallize method?
Finalize method is used to free the memory used by some unmanaged resources like window handles (HWND). It's similar to the destructor syntax in C#. The GC calls this method when it founds no more references to the object. But, In some cases we may need release the memory used by the resources explicitely.To release the memory explicitly we need to implement the Dispose method of IDisposable interface.

What is encapsulation ?
Encapsulation is the ability to hide the internal workings of an object's behavior and its data. For instance, let's say you have a object named Bike and this object has a method named start(). When you create an instance of a Bike object and call its start() method you are not worried about what happens to accomplish this, you just want to make sure the state of the bike is changed to 'running' afterwards. This kind of behavior hiding is encapsulation and it makes programming much easier.

How can you prevent your class to be inherated further?
By setting Sealed - Key word

```
public sealed class Planet
{
        //code goes here
}

class Moon:Planet
{
   //Not allowed as base class is sealed
}
```

What is GUID and why we need to use it and in what condition? How this is created.
A GUID is a 128-bit integer (16 bytes) that can be used across all computers and networks wherever a unique identifier is required. Such an identifier has a very low probability of being duplicated. Visual Studio .NET IDE has a utility under the tools menu to generate GUIDs.

Why do you need to serialize.?
We need to serialize the object,if you want to pass object from one computer/application domain to another.Process of converting complex objects into stream of bytes that can be persisted or transported.Namespace for serialization is System.Runtime.Serialization.The ISerializable interface allows you to make any class Serializable..NET framework features 2 serializing method.
1.Binary Serialization 2.XML Serialization

What is inline schema, how does it works?
Schemas can be included inside of XML file is called Inline Schemas.This is useful when it is inconvenient to physically seprate the schema and the XML document.A schema is an XML document that defines the structure, constraints, data types, and relationships of the elements that constitute the data contained inside the XML document or in another XML document.Schema can be an external file which uses the XSD or XDR extension called external schema. Inline schema can take place even when validation is turned off.

Describe the advantages of writing a managed code application instead of unmanaged one. What's involved in certain piece of code being managed?
"Advantage includes automatic garbage collection,memory management,security,type checking,versioning

Managed code is compiled for the .NET run-time environment. It runs in the Common Language Runtime (CLR), which is the heart of the .NET Framework. The CLR provides services such as security, memory management, and cross-language integration. Managed applications written to take advantage of the features of the CLR perform more efficiently and safely, and take better advantage of developers existing expertise in languages that support the .NET Framework.

Unmanaged code includes all code written before the .NET Framework was introduced—this includes code written to use COM, native Win32, and Visual Basic 6. Because it does not run inside the .NET environment, unmanaged code cannot make use of any .NET managed facilities."

What are multicast delegates ? give me an example ?
Delegate that can have more than one element in its invocation List.

```
using System;
namespace SampleMultiCastDelegate{
  class MultiCast{
    public delegate string strMultiCast(string s);
  }
}
```

MainClass defines the static methods having same signature as delegate.

```
using System;

namespace SampleMultiCastDelegate
{

  public class MainClass
  {
    public MainClass()
    {
    }

    public static string Jump(string s)
    {
      Console.WriteLine("Jump");
      return String.Empty;
    }

    public static string Run(string s)
    {
      Console.WriteLine("Run");
      return String.Empty;
    }

    public static string Walk(string s)
    {
      Console.WriteLine("Walk");
      return String.Empty;
    }
  }
}
```

The Main class:

```
using System;
using System.Threading;
namespace SampleMultiCastDelegate
{

  public class MainMultiCastDelegate
  {
    public static void Main()
    {
      MultiCast.strMultiCast  Run,Walk,Jump;

      MultiCast.strMultiCast    myDelegate;
```

```
      ///here mydelegate used the Combine method of
System.MulticastDelegate
     ///and the delegates combine

myDelegate=(MultiCast.strMultiCast)System.Delegate.Combine(Run,Walk);

   }
  }
}
```

Can a nested object be used in Serialization ?
Yes. If a class that is to be serialized contains references to objects of other classes, and if those classes have been marked as serializable, then their objects are serialized too.

Difference between int and int32 ?
Both are same. System.Int32 is a .NET class. Int is an alias name for System.Int32.

Describe the difference between a Thread and a Process?
A Process is an instance of an running application. And a thread is the Execution stream of the Process. A process can have multiple Thread. When a process starts a specific memory area is allocated to it. When there is multiple thread in a process, each thread gets a memory for storing the variables in it and plus they can access to the global variables which is common for all the thread. Eg.A Microsoft Word is a Application. When you open a word file,an instance of the Word starts and a process is allocated to this instance which has one thread.

What is the difference between an EXE and a DLL?
You can create an objects of Dll but not of the EXE.
Dll is an In-Process Component whereas EXE is an OUt-Process Component.
Exe is for single use whereas you can use Dll for multiple use.
Exe can be started as standalone where dll cannot be.

What is strong-typing versus weak-typing? Which is preferred? Why?
Strong typing implies that the types of variables involved in operations are associated to the variable, checked at compile-time, and require explicit conversion; weak typing implies that they are associated to the value, checked at run-time, and are implicitly converted as required. (Which is preferred is a disputable point, but I personally prefer strong typing because I like my errors to be found as soon as possible.)

What is a PID? How is it useful when troubleshooting a system?
PID is the process Id of the application in Windows. Whenever a process starts running in the Windows environment, it is associated with an individual process Id or PID.

The PID (Process ID) a unique number for each item on the Process Tab, Image Name list. How do you get the PID to appear? In Task Manger, select the View menu, then select columns and check PID (Process Identifier).

In Linux, PID is used to debug a process explicitly. However we cannot do this in a windows environment.

Microsoft has launched a SDK called as Microsoft Operations Management (MOM). This uses the PID to find out which dll's have been loaded by a process in the memory. This is essentially helpful in situations where the Process which has a memory leak is to be traced to a erring dll. Personally I have never used a PID, our Windows debugger does the things required to find out

What is the GAC? What problem does it solve?
Each computer where the common language runtime is installed has a machine-wide code cache called the global assembly cache. The global assembly cache stores assemblies that are to be shared by several applications on the computer. This area is typically the folder under windows or winnt in the machine.

All the assemblies that need to be shared across applications need to be done through the Global assembly Cache only. However it is not necessary to install assemblies into the global assembly cache to make them accessible to COM interop or unmanaged code.

There are several ways to deploy an assembly into the global assembly cache:
· Use an installer designed to work with the global assembly cache. This is the preferred option for installing assemblies into the global assembly cache.
· Use a developer tool called the Global Assembly Cache tool (Gacutil.exe), provided by the .NET Framework SDK.
· Use Windows Explorer to drag assemblies into the cache.

GAC solves the problem of DLL Hell and DLL versioning. Unlike earlier situations, GAC can hold two assemblies of the same name but different version. This ensures that the applications which access a particular assembly continue to access the same assembly even if another version of that assembly is installed on that machine.

Describe what an Interface is and how it's different from a Class.
An interface is a structure of code which is similar to a class. An interface is a prototype for a class and is useful from a logical design perspective. Interfaces provide a means to define the protocols for a class without worrying about the implementation details. The syntax for creating interfaces follows:

```
interface Identifier {
  InterfaceBody
}
```

Identifier is the name of the interface and InterfaceBody refers to the abstract methods and static final variables that make up the interface. Because it is assumed that all the methods in an interface are abstract, it isn't necessary to use the abstract keyword

An interface is a description of some of the members available from a class. In practice, the syntax typically looks similar to a class definition, except that there's no code defined for the methods — just their name, the arguments passed and the type of the value returned.
So what good is it? None by itself. But you create an interface so that classes will implement it.

But what does it mean to implement an interface. The interface acts as a contract or promise. If a class implements an interface, then it must have the properties and methods of the interface defined in the class. This is enforced by the compiler.

Broadly the differentiators between classes and interfaces is as follows
- Interface should not have any implementation.
- Interface can not create any instance.
- Interface should provide high level abstraction from the implementation.
- Interface can have multiple inheritances.
- Default access level of the interface is public.

What is the difference between XML Web Services using ASMX and .NET Remoting using SOAP?
ASP.NET Web services and .NET Remoting provide a full suite of design options for cross-process and cross-plaform communication in distributed applications. In general, ASP.NET Web services provide the highest levels of interoperability with full support for WSDL and SOAP over HTTP, while .NET Remoting is designed for common language runtime type-system fidelity and supports additional data format and communication channels. Hence if we looking cross-platform communication than web services is the choice coz for .NET remoting .Net framework is requried which may or may not present for the other platform.

Serialization and Metadata
ASP.NET Web services rely on the System.Xml.Serialization.XmlSerializer

class to marshal data to and from SOAP messages at runtime. For metadata, they generate WSDL and XSD definitions that describe what their messages contain. The reliance on pure WSDL and XSD makes ASP.NET Web services metadata portable; it expresses data structures in a way that other Web service toolkits on different platforms and with different programming models can understand. In some cases, this imposes constraints on the types you can expose from a Web service—XmlSerializer will only marshal things that can be expressed in XSD. Specifically, XmlSerializer will not marshal object graphs and it has limited support for container types.

.NET Remoting relies on the pluggable implementations of the IFormatter interface used by the System.Runtime.Serialization engine to marshal data to and from messages. There are two standard formatters, System.Runtime.Serialization.Formatters.Binary.BinaryFormatter and System.Runtime.Serialization.Formatters.Soap.SoapFormatter. The BinaryFormatter and SoapFormatter, as the names suggest, marshal types in binary and SOAP format respectively. For metadata, .NET Remoting relies on the common language runtime assemblies, which contain all the relevant information about the data types they implement, and expose it via reflection. The reliance on the assemblies for metadata makes it easy to preserve the full runtime type-system fidelity. As a result, when the .NET Remoting plumbing marshals data, it includes all of a class's public and private members; handles object graphs correctly; and supports all container types (e.g., System.Collections.Hashtable). However, the reliance on runtime metadata also limits the reach of a .NET Remoting system—a client has to understand .NET constructs in order to communicate with a .NET Remoting endpoint. In addition to pluggable formatters, the .NET Remoting layer supports pluggable channels, which abstract away the details of how messages are sent. There are two standard channels, one for raw TCP and one for HTTP. Messages can be sent over either channel independent of format.

Distributed Application Design: ASP.NET Web Services vs. .NET Remoting
ASP.NET Web services favor the XML Schema type system, and provide a simple programming model with broad cross-platform reach. .NET Remoting favors the runtime type system, and provides a more complex programming model with much more limited reach. This essential difference is the primary factor in determining which technology to use. However, there are a wide range of other design factors, including transport protocols, host processes, security, performance, state management, and support for transactions to consider as well.

Security
Since ASP.NET Web services rely on HTTP, they integrate with the standard Internet security infrastructure. ASP.NET leverages the security features available with IIS to provide strong support for standard HTTP authentication schemes including Basic, Digest, digital certificates, and even Microsoft® .NET Passport. (You can also use Windows Integrated authentication, but

only for clients in a trusted domain.) One advantage of using the available HTTP authentication schemes is that no code change is required in a Web service; IIS performs authentication before the ASP.NET Web services are called. ASP.NET also provides support for .NET Passport-based authentication and other custom authentication schemes. ASP.NET supports access control based on target URLs, and by integrating with the .NET code access security (CAS) infrastructure. SSL can be used to ensure private communication over the wire.

Although these standard transport-level techniques to secure Web services are quite effective, they only go so far. In complex scenarios involving multiple Web services in different trust domains, you have to build custom ad hoc solutions. Microsoft and others are working on a set of security specifications that build on the extensibility of SOAP messages to offer message-level security capabilities. One of these is the XML Web Services Security Language (WS-Security), which defines a framework for message-level credential transfer, message integrity, and message confidentiality.

As noted in the previous section, the .NET Remoting plumbing does not secure cross-process invocations in the general case. A .NET Remoting endpoint hosted in IIS with ASP.NET can leverage all the same security features available to ASP.NET Web services, including support for secure communication over the wire using SSL. If you are using the TCP channel or the HTTP channel hosted in processes other than aspnet_wp.exe, you have to implement authentication, authorization and privacy mechanisms yourself.

One additional security concern is the ability to execute code from a semi-trusted environment without having to change the default security policy. ASP.NET Web Services client proxies work in these environments, but .NET Remoting proxies do not. In order to use a .NET Remoting proxy from a semi-trusted environment, you need a special serialization permission that is not given to code loaded from your intranet or the Internet by default. If you want to use a .NET Remoting client from within a semi-trusted environment, you have to alter the default security policy for code loaded from those zones. In situations where you are connecting to systems from clients running in a sandbox—like a downloaded Windows Forms application, for instance—ASP.NET Web Services are a simpler choice because security policy changes are not required.

Conceptually, what is the difference between early-binding and late-binding?
Early binding – Binding at Compile Time
Late Binding – Binding at Run Time

Early binding implies that the class of the called object is known at compile-time; late-binding implies that the class is not known until run-time, such as a call through an interface or via Reflection.

Early binding is the preferred method. It is the best performer because your application binds directly to the address of the function being called and there is no extra overhead in doing a run-time lookup. In terms of overall execution speed, it is at least twice as fast as late binding.

Early binding also provides type safety. When you have a reference set to the component's type library, Visual Basic provides IntelliSense support to help you code each function correctly. Visual Basic also warns you if the data type of a parameter or return value is incorrect, saving a lot of time when writing and debugging code.

Late binding is still useful in situations where the exact interface of an object is not known at design-time. If your application seeks to talk with multiple unknown servers or needs to invoke functions by name (using the Visual Basic 6.0 CallByName function for example) then you need to use late binding. Late binding is also useful to work around compatibility problems between multiple versions of a component that has improperly modified or adapted its interface between versions.

What is an Asssembly Qualified Name? Is it a filename? How is it different?
An assembly qualified name isn't the filename of the assembly; it's the internal name of the assembly combined with the assembly version, culture, and public key, thus making it unique.

e.g. (""System.Xml.XmlDocument, System.Xml, Version=1.0.3300.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"")

How is a strongly-named assembly different from one that isn't strongly-named?
Strong names are used to enable the stricter naming requirements associated with shared assemblies. These strong names are created by a .NET utility – sn.exe

Strong names have three goals:
· Name uniqueness. Shared assemblies must have names that are globally unique.
· Prevent name spoofing. Developers don't want someone else releasing a subsequent version of one of your assemblies and falsely claim it came from you, either by accident or intentionally.
· Provide identity on reference. When resolving a reference to an assembly, strong names are used to guarantee the assembly that is loaded came from the expected publisher.

Strong names are implemented using standard public key cryptography. In general, the process works as follows: The author of an assembly generates a key pair (or uses an existing one), signs the file containing the manifest with the private key, and makes the public key available to callers. When

references are made to the assembly, the caller records the public key corresponding to the private key used to generate the strong name.

Weak named assemblies are not suitable to be added in GAC and shared. It is essential for an assembly to be strong named.

Strong naming prevents tampering and enables assemblies to be placed in the GAC alongside other assemblies of the same name.

How does the generational garbage collector in the .NET CLR manage object lifetime? What is non-deterministic finalization?
The hugely simplistic version is that every time it garbage-collects, it starts by assuming everything to be garbage, then goes through and builds a list of everything reachable. Those become not-garbage, everything else doesn't, and gets thrown away. What makes it generational is that every time an object goes through this process and survives, it is noted as being a member of an older generation (up to 2, right now). When the garbage-collector is trying to free memory, it starts with the lowest generation (0) and only works up to higher ones if it can't free up enough space, on the grounds that shorter-lived objects are more likely to have been freed than longer-lived ones.

Non-deterministic finalization implies that the destructor (if any) of an object will not necessarily be run (nor its memory cleaned up, but that's a relatively minor issue) immediately upon its going out of scope. Instead, it will wait until first the garbage collector gets around to finding it, and then the finalisation queue empties down to it; and if the process ends before this happens, it may not be finalised at all. (Although the operating system will usually clean up any process-external resources left open - note the usually there, especially as the exceptions tend to hurt a lot.)

What is the difference between Finalize() and Dispose()?
Dispose() is called by the user of an object to indicate that he is finished with it, enabling that object to release any unmanaged resources it holds. Finalize() is called by the run-time to allow an object which has not had Dispose() called on it to do the same. However, Dispose() operates determinalistically, whereas there is no guarantee that Finalize() will be called immediately when an object goes out of scope - or indeed at all, if the program ends before that object is GCed - and as such Dispose() is generally preferred.

How is the using() pattern useful? What is IDisposable? How does it support deterministic finalization?
The using() pattern is useful because it ensures that Dispose() will always be called when a disposable object (defined as one that implements IDisposable, and thus the Dispose() method) goes out of scope, even if it does so by an exception being thrown, and thus that resources are always released.

What does this useful command line do? tasklist /m "mscor*"
Lists all the applications and associated tasks/process currently running on the system with a module whose name begins "mscor" loaded into them; which in nearly all cases means "all the .NET processes".

What's wrong with a line like this? DateTime.Parse(myString);
Therez nothing wrong with this declaration.Converts the specified string representation of a date and time to its DateTime equivalent.But If the string is not a valid DateTime,It throws an exception.

What are PDBs? Where must they be located for debugging to work?
A program database (PDB) files holds debugging and project state information that allows incremental linking of debug configuration of your program.There are several different types of symbolic debugging information. The default type for Microsoft compiler is the so-called PDB file. The compiler setting for creating this file is /Zi, or /ZI for C/C++(which creates a PDB file with additional information that enables a feature called ""Edit and Continue"") or a Visual Basic/C#/JScript .NET program with /debug.

A PDB file is a separate file, placed by default in the Debug project subdirectory, that has the same name as the executable file with the extension .pdb. Note that the Visual C++ compiler by default creates an additional PDB file called VC60.pdb for VisulaC++6.0 and VC70.PDB file for VisulaC++7.0. The compiler creates this file during compilation of the source code, when the compiler isn't aware of the final name of the executable. The linker can merge this temporary PDB file into the main one if you tell it to, but it won't do it by default. The PDB file can be useful to display the detailed stack trace with source files and line numbers.

What is FullTrust? Do GAC'ed assemblies have FullTrust?
Before the .NET Framework existed, Windows had two levels of trust for downloaded code. This old model was a binary trust model. You only had two choices: Full Trust, and No Trust. The code could either do anything you could do, or it wouldn't run at all.

The permission sets in .NET include FullTrust, SkipVerification, Execution, Nothing, LocalIntranet, Internet and Everything. Full Trust Grants unrestricted permissions to system resources. Fully trusted code run by a normal, nonprivileged user cannot do administrative tasks, but can access any resources the user can access, and do anything the user can do. From a security standpoint, you can think of fully trusted code as being similar to native, unmanaged code, like a traditional ActiveX control.
GAC assemblies are granted FullTrust. In v1.0 and 1.1, the fact that assemblies in the GAC seem to always get a FullTrust grant is actually a side effect of the fact that the GAC lives on the local machine. If anyone were to lock down the security policy by changing the grant set of the local machine to something less than FullTrust, and if your assembly did not get extra

permission from some other code group, it would no longer have FullTrust even though it lives in the GAC.

What does this do? gacutil /l | find /i "Corillian"
The Global Assembly Cache tool allows you to view and manipulate the contents of the global assembly cache and download cache.The tool comes with various optional params to do that.
""/l"" option Lists the contents of the global assembly cache. If you specify the assemblyName parameter(/l [assemblyName]), the tool lists only the assemblies matching that name.

What does this do .. sn -t foo.dll ?
Sn -t option displays the token for the public key stored in infile. The contents of infile must be previously generated using -p.
Sn.exe computes the token using a hash function from the public key. To save space, the common language runtime stores public key tokens in the manifest as part of a reference to another assembly when it records a dependency to an assembly that has a strong name. The -tp option displays the public key in addition to the token.

How do you generate a strong name?
.NET provides an utility called strong name tool. You can run this toolfrom the VS.NET command prompt to generate a strong name with an option "-k" and providing the strong key file name. i.e. sn- -k < file-name >

What is the difference between a Debug and Release build? Is there a significant speed difference? Why or why not?
The Debug build is the program compiled with full symbolic debug information and no optimization. The Release build is the program compiled employing optimization and contains no symbolic debug information. These settings can be changed as per need from Project Configuration properties. The release runs faster since it does not have any debug symbols and is optimized.

Explain the use of virtual, sealed, override, and abstract.
Abstract: The keyword can be applied for a class or method.
1. Class: If we use abstract keyword for a class it makes the
class an abstract class, which means it cant be instantiated. Though
it is not nessacary to make all the method within the abstract class to be virtual. ie, Abstract class can have concrete methods
2. Method: If we make a method as abstract, we dont need to provide implementation
of the method in the class but the derived class need to implement/override this method.

Sealed: It can be applied on a class and methods. It stops the type from further derivation i.e no one can derive class
from a sealed class,ie A sealed class cannot be inherited.A sealed class

cannot be a abstract class.A compile time error is thrown if you try to specify sealed class as a base class.
When an instance method declaration includes a sealed modifier, that method is said to be a sealed method. If an instance method declaration includes the sealed modifier, it must also include the override modifier. Use of the sealed modifier prevents a derived class from further overriding the method For Egs: sealed override public void Sample() { Console.WriteLine("Sealed Method"); }

Virtual & Override: Virtual & Override keywords provides runtime polymorphism. A base class can make some of its methods
as virtual which allows the derived class a chance to override the base class implementation by using override keyword.

For e.g. class Shape

```
  {
  int a
  public virtual void Display()
  {
   Console.WriteLine("Shape");
  }
 }

 class Rectangle:Shape
 {
  public override void Display()
  {
   Console.WriteLine("Derived");
  }
 }
```

Explain the importance and use of each, Version, Culture and PublicKeyToken for an assembly.
This three alongwith name of the assembly provide a strong name or fully qualified name to the assembly. When a assebly is referenced with all three.

PublicKeyToken: Each assembly can have a public key embedded in its manifest that identifies the developer. This ensures that once the assembly ships, no one can modify the code or other resources contained in the assembly.

Culture: Specifies which culture the assembly supports

Version: The version number of the assembly.It is of the following form major.minor.build.revision.

Explain the differences between public, protected, private and internal.
These all are access modifier and they governs the access level. They can be

applied to class, methods, fields.

Public: Allows class, methods, fields to be accessible from anywhere i.e. within and outside an assembly.
Private: When applied to field and method allows to be accessible within a class.
Protected: Similar to private but can be accessed by members of derived class also.
Internal: They are public within the assembly i.e. they can be accessed by anyone within an assembly but outside assembly they are not visible.

What is the difference between typeof(foo) and myFoo.GetType()?
Typeof is operator which applied to a object returns System.Type object. Typeof cannot be overloaded white GetType has lot of overloads.GetType is a method which also returns System.Type of an object. GetType is used to get the runtime type of the object.

Example from MSDN showing Gettype used to retrive type at untime:-

```
public class MyBaseClass: Object {
}

public class MyDerivedClass: MyBaseClass {
}

public class Test {

  public static void Main() {
    MyBaseClass myBase = new MyBaseClass();
    MyDerivedClass myDerived = new MyDerivedClass();
    object o = myDerived;
    MyBaseClass b = myDerived;

    Console.WriteLine("mybase: Type is {0}", myBase.GetType());
    Console.WriteLine("myDerived: Type is {0}", myDerived.GetType());
    Console.WriteLine("object o = myDerived: Type is {0}", o.GetType());
    Console.WriteLine("MyBaseClass b = myDerived: Type is {0}",
b.GetType());
  }
}
```

```
/*

This code produces the following output.

mybase: Type is MyBaseClass
myDerived: Type is MyDerivedClass
object o = myDerived: Type is MyDerivedClass
MyBaseClass b = myDerived: Type is MyDerivedClass

*/
```

Can "this" be used within a static method?
No 'This' cannot be used in a static method. As only static variables/methods can be used in a static method.

What is the purpose of XML Namespaces?
An XML Namespace is a collection of element types and attribute names. It consists of 2 parts
1) The first part is the URI used to identify the namespace
2) The second part is the element type or attribute name itself.
Together they form a unique name. The various purpose of XML Namespace are

1. Combine fragments from different documents without any naming conflicts. (See example below.)
2. Write reusable code modules that can be invoked for specific elements and attributes. Universally unique names guarantee that
such modules are invoked only for the correct elements and attributes.
3. Define elements and attributes that can be reused in other schemas or instance documents without fear of name collisions. For
example, you might use XHTML elements in a parts catalog to provide part descriptions. Or you might use the nil attribute
defined in XML Schemas to indicate a missing value.

```
< Department >
    < Name >DVS1< /Name >
    < addr:Address xmlns:addr="http://www.tu-darmstadt.de/ito/addresses"
>
      < addr:Street >Wilhelminenstr. 7< /addr:Street >
      < addr:City >Darmstadt< /addr:City >
      < addr:State >Hessen< /addr:State >
      < addr:Country >Germany< /addr:Country >
      < addr:PostalCode >D-64285< /addr:PostalCode >
    < /addr:Address >
    < serv:Server xmlns:serv="http://www.tu-darmstadt.de/ito/servers" >
      < serv:Name >OurWebServer< /serv:Name >
      < serv:Address >123.45.67.8< /serv:Address >
```

```
    < /serv:Server >
  < /Department >
```

What is difference between MetaData and Manifest ?
Metadata and Manifest forms an integral part of an assembly(dll / exe) in .net framework .
Out of which Metadata is a mandatory component , which as the name suggests gives the details about various components of IL code viz : Methods , properties , fields , class etc.

Essentially Metadata maintains details in form of tables like Methods Metadata tables , Properties Metadata tables , which maintains the list of given type and other details like access specifier , return type etc.

Now Manifest is a part of metadata only , fully called as “manifest metadata tables” , it contains the details of the references needed by the assembly of any other external assembly / type , it could be a custom assembly or standard System namespace .

Now for an assembly that can independently exists and used in the .Net world both the things (Metadata with Manifest) are mandatory , so that it can be fully described assembly and can be ported anywhere without any system dependency . Essentially .Net framework can read all assembly related information from assembly itself at runtime .

But for .Net modules , that can’t be used independently , until they are being packaged as a part of an assembly , they don’t contain Manifest but their complete structure is defined by their respective metadata .

Ultimately . .Net modules use Manifest Metadata tables of parent assembly which contain them .

What is the use of Internal keyword?
Internal keyword is one of the access specifier available in .Net framework , that makes a type visible in a given assembly , for e.g : a single dll can contain multiple modules , essentially a multi file assembly , but it forms a single binary component , so any type with internal keyword will be visible throughout the assembly and can be used in any of the modules .

What actually happes when you add a something to arraylistcollection ?
Following things will happen :

Arraylist is a dynamic array class in c# in System.Collections namespace derived from interfaces – ICollection , IList , ICloneable , IConvertible . It terms of in memory structure following is the implementation .

a. Check up the total space if there's any free space on the declared list .
b. If yes add the new item and increase count by 1 .
c. If No Copy the whole thing to a temporary Array of Last Max. Size .
d. Create new Array with size (Last Array Size + Increase Value)
e. Copy back values from temp and reference this new array as original array .
f. Must doing Method updates too , need to check it up .

What is Boxing and unboxing? Does it occure automaatically or u need to write code to box and unbox?
Boxing – Process of converting a System.ValueType to Reference Type , Mostly base class System.Object type and allocating it memory on Heap .Reverse is unboxing , but can only be done with prior boxed variables.

Boxing is always implicit but Unboxing needs to be explicitly done via casting , thus ensuring the value type contained inside .

How Boxing and unboxing occures in memory?
Boxing converts value type to reference type , thus allocating memory on Heap . Unboxing converts already boxed reference types to value types through explicit casting , thus allocating memory on stack .

Why only boxed types can be unboxed?
Unboxing is the process of converting a Reference type variable to Value type and thus allocating memory on the stack . It happens only to those Reference type variables that have been earlier created by Boxing of a Value Type , therefore internally they contain a value type , which can be obtained through explicit casting . For any other Reference type , they don't internally contain a Value type to Unboxed via explicit casting . This is why only boxed types can be unboxed .

Assembly Questions

1. **How is the DLL Hell problem solved in .NET?**
   Assembly versioning allows the application to specify not only the library it needs to run (which was available under Win32), but also the version of the assembly.

2. **What are the ways to deploy an assembly?**
   An MSI installer, a CAB archive, and XCOPY command.

3. **What is a satellite assembly?**
   When you write a multilingual or multi-cultural application in .NET, and want to distribute the core application separately from the localized modules, the localized assemblies that modify the core application are called satellite assemblies.

4. **What namespaces are necessary to create a localized application?**
   System.Globalization and System.Resources.

5. **What is the smallest unit of execution in .NET?**
   an Assembly.

6. **When should you call the garbage collector in .NET?**
   As a good rule, you should not call the garbage collector. However, you could call the garbage collector when you are done using a large object (or set of objects) to force the garbage collector to dispose of those very large objects from memory. However, this is usually not a good practice.

7. **How do you convert a value-type to a reference-type?**
   Use Boxing.

8. **What happens in memory when you Box and Unbox a value-type?**
   Boxing converts a value-type to a reference-type, thus storing the object on the heap. Unboxing converts a reference-type to a value-type, thus storing the value on the stack.

1. **How many languages .NET is supporting now?** - When .NET was introduced it came with several languages. VB.NET, C#, COBOL and Perl, etc. The site DotNetLanguages.Net says 44 languages are supported.
2. **How is .NET able to support multiple languages?** - a language should comply with the Common Language Runtime standard to become a .NET language. In .NET, code is compiled to Microsoft Intermediate Language (MSIL for short). This is called as Managed Code. This Managed code is run in .NET environment. So after compilation to this IL the language is not a barrier. A code can call or use a function written in another language.
3. **How ASP .NET different from ASP?** - Scripting is separated from the HTML, Code is compiled as a DLL, these DLLs can be executed on the server.
4. **Resource Files: How to use the resource files, how to know which language to use?**
5. **What is smart navigation?** - The cursor position is maintained when the page gets refreshed due to the server side validation and the page gets refreshed.
6. **What is view state?** - The web is stateless. But in ASP.NET, the state of a page is maintained in the in the page itself automatically. How? The values are encrypted and saved in hidden controls. this is done automatically by the ASP.NET. This can be switched off / on for a single control
7. **Explain the life cycle of an ASP .NET page.**
8. **How do you validate the controls in an ASP .NET page?** - Using special validation controls that are meant for this. We have Range Validator, Email Validator.
9. **Can the validation be done in the server side? Or this can be done only in the Client side?** - Client side is done by default. Server side validation is also possible. We can switch off the client side and server side can be done.
10. **How to manage pagination in a page?** - Using pagination option in DataGrid control. We have to set the number of records for a page, then it takes care of pagination by itself.
11. **What is ADO .NET and what is difference between ADO and ADO.NET?** - ADO.NET is stateless mechanism. I can treat the ADO.Net as a separate in-memory database where in I can use relationships between the tables and select insert and updates to the database. I can update the actual database as a batch

WEB FORMS

What base class do all Web Forms inherit from?
System.Windows.Forms.Form

What is the difference between Debug.Write and Trace.Write? When should each be used?
The Debug.Write call won't be compiled when the DEBUGsymbol is not defined (when doing a release build). Trace.Write calls will be compiled. Debug.Write is for information you want only in debug builds, Trace.Write is for when you want it in release build as well.

Difference between Anchor and Dock Properties?
Dock Property->Gets or sets which edge of the parent container a control is docked to. A control can be docked to one edge of its parent container or can be docked to all edges and fill the parent container. For example, if you set this property to DockStyle.Left, the left edge of the
control will be docked to the left edge of its parent control. Additionally, the docked edge of the control is resized to match that of its container control.
Anchor Property->Gets or sets which edges of the control are anchored to the edges of its container. A control can be anchored to one or more edges of its parent container. Anchoring a control to its parent ensures that the anchored edges remain in the same position relative to the edges of the parent container when the parent container is resized.

When would you use ErrorProvider control?
ErrorProvider control is used in Windows Forms application. It is like Validation Control for ASP.NET pages. ErrorProvider control is used to provide validations in Windows forms and display user friendly messages to the user if the validation fails.
E.g
If we went to validate the textBox1 should be empty, then we can validate as below
1). You need to place the errorprovide control on the form

```
private void textBox1_Validating(object sender,
System.ComponentModel.CancelEventArgs e)
{
ValidateName();
}
private bool ValidateName()
{
bool bStatus = true;
if (textBox1.Text == "")
{
errorProvider1.SetError (textBox1,"Please enter your Name");
bStatus = false;
```

```
}
else
errorProvider1.SetError (textBox1,"");
return bStatus;
}
```
it check the textBox1 is empty . If it is empty, then a message Please enter your name is displayed.

Can you write a class without specifying namespace? Which namespace does it belong to by default??
Yes, you can, then the class belongs to global namespace which has no name. For commercial products, naturally, you wouldn't want global namespace.

You are designing a GUI application with a windows and several widgets on it. The user then resizes the app window and sees a lot of grey space, while the widgets stay in place. What's the problem?
One should use anchoring for correct resizing. Otherwise the default property of a widget on a form is top-left, so it stays at the same location when resized.

How can you save the desired properties of Windows Forms application?
.config files in .NET are supported through the API to allow storing and retrieving information. They are nothing more than simple XML files, sort of like what .ini files were before for Win32 apps.

So how do you retrieve the customized properties of a .NET application from XML .config file?
Initialize an instance of AppSettingsReader class. Call the GetValue method of AppSettingsReader class, passing in the name of the property and the type expected. Assign the result to the appropriate variable.

Can you automate this process?
In Visual Studio yes, use Dynamic Properties for automatic .config creation, storage and retrieval.

My progress bar freezes up and dialog window shows blank, when an intensive background process takes over.
Yes, you should've multi-threaded your GUI, with taskbar and main form being one thread, and the background process being the other.

What's the safest way to deploy a Windows Forms app?
Web deployment: the user always downloads the latest version of the code, the program runs within security sandbox, properly written app will not require additional security privileges.

Why is it not a good idea to insert code into InitializeComponent method when working with Visual Studio?
The designer will likely through it away, most of the code inside InitializeComponent is auto-generated.

What's the difference between WindowsDefaultLocation and WindowsDefaultBounds?
WindowsDefaultLocation tells the form to start up at a location selected by OS, but with internally specified size. WindowsDefaultBounds delegates both size and starting position choices to the OS.

What's the difference between Move and LocationChanged? Resize and SizeChanged?
Both methods do the same, Move and Resize are the names adopted from VB to ease migration to C#.

How would you create a non-rectangular window, let's say an ellipse?
Create a rectangular form, set the TransparencyKey property to the same value as BackColor, which will effectively make the background of the form transparent. Then set the FormBorderStyle to FormBorderStyle.None, which will remove the contour and contents of the form.

How do you create a separator in the Menu Designer?
A hyphen '-' would do it. Also, an ampersand '&\' would underline the next letter.

How's anchoring different from docking?
Anchoring treats the component as having the absolute size and adjusts its location relative to the parent form. Docking treats the component location as absolute and disregards the component size. So if a status bar must always be at the bottom no matter what, use docking. If a button should be on the top right, but change its position with the form being resized, use anchoring.

How do you trigger the Paint event in System.Drawing?
Invalidate the current form, the OS will take care of repainting. The Update method forces the repaint.

With these events, why wouldn't Microsoft combine Invalidate and Paint, so that you wouldn't have to tell it to repaint, and then to force it to repaint?
Painting is the slowest thing the OS does, so usually telling it to repaint, but not forcing it allows for the process to take place in the background.

How can you assign an RGB color to a System.Drawing.Color object?
Call the static method FromArgb of this class and pass it the RGB values.

What class does Icon derive from?
Isn't it just a Bitmap with a wrapper name around it? No, Icon lives in System.Drawing namespace. It's not a Bitmap by default, and is treated separately by .NET. However, you can use ToBitmap method to get a valid Bitmap object from a valid Icon object.

Before in my VB app I would just load the icons from DLL. How can I load the icons provided by .NET dynamically?
By using System.Drawing.SystemIcons class, for example System.Drawing.SystemIcons.Warning produces an Icon with a warning sign in it.

When displaying fonts, what's the difference between pixels, points and ems?
A pixel is the lowest-resolution dot the computer monitor supports. Its size depends on user's settings and monitor size. A point is always 1/72 of an inch. An em is the number of pixels that it takes to display the letter M

ADO.NET and Database Questions

1. **What is the role of the DataReader class in ADO.NET connections?**
   It returns a read-only, forward-only rowset from the data source. A DataReader provides fast access when a forward-only sequential read is needed.
2. **What are advantages and disadvantages of Microsoft-provided data provider classes in ADO.NET?**
   SQLServer.NET data provider is high-speed and robust, but requires SQL Server license purchased from Microsoft. OLE-DB.NET is universal for accessing other sources, like Oracle, DB2, Microsoft Access and Informix. OLE-DB.NET is a .NET layer on top of the OLE layer, so it's not as fastest and efficient as SqlServer.NET.
3. **What is the wildcard character in SQL?**
   Let's say you want to query database with LIKE for all employees whose name starts with La. The wildcard character is %, the proper query with LIKE would involve 'La%'.

4. **Explain ACID rule of thumb for transactions.**
   A transaction must be:
   1. Atomic - it is one unit of work and does not dependent on previous and following transactions.
   2. Consistent - data is either committed or roll back, no "in-between" case where something has been updated and something hasn't.

3. Isolated - no transaction sees the intermediate results of the current transaction).
4. Durable - the values persist if the data had been committed even if the system crashes right after.

5. **What connections does Microsoft SQL Server support?**
   Windows Authentication (via Active Directory) and SQL Server authentication (via Microsoft SQL Server username and password).

6. **Between Windows Authentication and SQL Server Authentication, which one is trusted and which one is untrusted?**
   Windows Authentication is trusted because the username and password are checked with the Active Directory, the SQL Server authentication is untrusted, since SQL Server is the only verifier participating in the transaction.

7. **What does the Initial Catalog parameter define in the connection string?**
   The database name to connect to.

8. **What does the Dispose method do with the connection object?**
   Deletes it from the memory.
   To Do: answer better. The current answer is not entirely correct.

9. **What is a pre-requisite for connection pooling?**
   Multiple processes must agree that they will share the same connection, where every parameter is the same, including the security settings. The connection string must be identical.

Debugging and Testing Questions

1. **What debugging tools come with the .NET SDK?**
   1. CorDBG – command-line debugger. To use CorDbg, you must compile the original C# file using the /debug switch.
   2. DbgCLR – graphic debugger. Visual Studio .NET uses the DbgCLR.

2. **What does assert() method do?**
   In debug compilation, assert takes in a Boolean condition as a parameter, and shows the error dialog if the condition is false. The program proceeds without any interruption if the condition is true.

3. **What's the difference between the Debug class and Trace class?**
   Documentation looks the same. Use Debug class for debug builds, use Trace class for both debug and release builds.

4. **Why are there five tracing levels in System.Diagnostics.TraceSwitcher?**
   The tracing dumps can be quite verbose. For applications that are constantly running you run the risk of overloading the machine and the hard drive. Five levels range from None to Verbose, allowing you to fine-tune the tracing activities.

5. **Where is the output of TextWriterTraceListener redirected?**
   To the Console or a text file depending on the parameter passed to the constructor.

6. **How do you debug an ASP.NET Web application?**
   Attach the aspnet_wp.exe process to the DbgClr debugger.

7. **What are three test cases you should go through in unit testing?**
   1. Positive test cases (correct data, correct output).
   2. Negative test cases (broken or missing data, proper handling).
   3. Exception test cases (exceptions are thrown and caught properly).

8. **Can you change the value of a variable while debugging a C# application?**
   Yes. If you are debugging via Visual Studio.NET, just go to Immediate window.

What is view state and use of it?
The current property settings of an ASP.NET page and those of any ASP.NET server controls contained within the page. ASP.NET can detect when a form is requested for the first time versus when the form is posted (sent to the server), which allows you to program accordingly.

What are user controls and custom controls?
Custom controls:
A control authored by a user or a third-party software vendor that does not belong to the .NET Framework class library. This is a generic term that includes user controls. A custom server control is used in Web Forms (ASP.NET pages). A custom client control is used in Windows Forms applications.

User Controls:
In ASP.NET: A user-authored server control that enables an ASP.NET page to be re-used as a server control. An ASP.NET user control is authored declaratively and persisted as a text file with an .ascx extension. The ASP.NET page framework compiles a user control on the fly to a class that derives from the System.Web.UI.UserControl class.

What are the validation controls?
A set of server controls included with ASP.NET that test user input in HTML and Web server controls for programmer-defined requirements. Validation controls perform input checking in server code. If the user is working with a browser that supports DHTML, the validation controls can also perform validation using client script.

What's the difference between Response.Write() andResponse.Output.Write()?
The latter one allows you to write formattedoutput.

What methods are fired during the page load? Init()
When the page is instantiated, Load() - when the page is loaded into server memory,PreRender () - the brief moment before the page is displayed to the user as HTML, Unload() - when page finishes loading.

Where does the Web page belong in the .NET Framework class hierarchy?
System.Web.UI.Page

Where do you store the information about the user's locale?
System.Web.UI.Page.Culture

What's the difference between Codebehind="MyCode.aspx.cs" and Src="MyCode.aspx.cs"?
CodeBehind is relevant to Visual Studio.NET only.

What's a bubbled event?
When you have a complex control, likeDataGrid, writing an event processing routine for each object (cell, button,row, etc.) is quite tedious. The controls can bubble up their eventhandlers, allowing the main DataGrid event handler to take care of its constituents.
Suppose you want a certain ASP.NET function executed on MouseOver over a certain button.

Where do you add an event handler?
It's the Attributesproperty, the Add function inside that property.
e.g. btnSubmit.Attributes.Add("onMouseOver","someClientCode();")

What data type does the RangeValidator control support?
Integer,String and Date.

What are the different types of caching?
Caching is a technique widely used in computing to increase performance by keeping frequently accessed or expensive data in memory. In context of web application, caching is used to retain the pages or data across HTTP requests and reuse them without the expense of recreating them.ASP.NET has 3 kinds of caching strategiesOutput CachingFragment CachingData

CachingOutput Caching: Caches the dynamic output generated by a request. Some times it is useful to cache the output of a website even for a minute, which will result in a better performance. For caching the whole page the page should have OutputCache directive.<%@ OutputCache Duration="60" VaryByParam="state" %>

Fragment Caching: Caches the portion of the page generated by the request. Some times it is not practical to cache the entire page, in such cases we can cache a portion of page<%@ OutputCache Duration="120" VaryByParam="CategoryID;SelectedID"%>

Data Caching: Caches the objects programmatically. For data caching asp.net provides a cache object for eg: cache["States"] = dsStates;

What do you mean by authentication and authorization?
Authentication is the process of validating a user on the credentials (username and password) and authorization performs after authentication. After Authentication a user will be verified for performing the various tasks, It access is limited it is known as authorization.

What are different types of directives in .NET?
@Page: Defines page-specific attributes used by the ASP.NET page parser and compiler. Can be included only in .aspx files <%@ Page AspCompat="TRUE" language="C#" %>
@Control:Defines control-specific attributes used by the ASP.NET page parser and compiler. Can be included only in .ascx files. <%@ Control Language="VB" EnableViewState="false" %>
@Import: Explicitly imports a namespace into a page or user control. The Import directive cannot have more than one namespace attribute. To import multiple namespaces, use multiple @Import directives. <% @ Import Namespace="System.web" %>
@Implements: Indicates that the current page or user control implements the specified .NET framework interface.<%@ Implements Interface="System.Web.UI.IPostBackEventHandler" %>
@Register: Associates aliases with namespaces and class names for concise notation in custom server control syntax.<%@ Register Tagprefix="Acme" Tagname="AdRotator" Src="AdRotator.ascx" %>
@Assembly: Links an assembly to the current page during compilation, making all the assembly's classes and interfaces available for use on the page. <%@ Assembly Name="MyAssembly" %><%@ Assembly Src="MySource.vb" %>
@OutputCache: Declaratively controls the output caching policies of an ASP.NET page or a user control contained in a page<%@ OutputCache Duration="#ofseconds" Location="Any | Client | Downstream | Server | None" Shared="True | False" VaryByControl="controlname" VaryByCustom="browser | customstring" VaryByHeader="headers" VaryByParam="parametername" %>
@Reference: Declaratively indicates that another user control or page

source file should be dynamically compiled and linked against the page in which this directive is declared.

How do I debug an ASP.NET application that wasn't written with Visual Studio.NET and that doesn't use code-behind?

Start the DbgClr debugger that comes with the .NET Framework SDK, open the file containing the code you want to debug, and set your breakpoints. Start the ASP.NET application. Go back to DbgClr, choose Debug Processes from the Tools menu, and select aspnet_wp.exe from the list of processes. (If aspnet_wp.exe doesn't appear in the list,check the "Show system processes" box.) Click the Attach button to attach to aspnet_wp.exe and begin debugging.

Be sure to enable debugging in the ASPX file before debugging it with DbgClr. You can enable tell ASP.NET to build debug executables by placing a

<%@ Page Debug="true" %> statement at the top of an ASPX file or a <COMPILATION debug="true" />statement in a Web.config file.

Can a user browsing my Web site read my Web.config or Global.asax files?

No. The <HTTPHANDLERS>section of Machine.config, which holds the master configuration settings for ASP.NET, contains entries that map ASAX files, CONFIG files, and selected other file types to an HTTP handler named HttpForbiddenHandler, which fails attempts to retrieve the associated file. You can modify it by editing Machine.config or including an section in a local Web.config file.

What's the difference between Page.RegisterClientScriptBlock and Page.RegisterStartupScript?

RegisterClientScriptBlock is for returning blocks of client-side script containing functions. RegisterStartupScript is for returning blocks of client-script not packaged in functions-in other words, code that's to execute when the page is loaded. The latter positions script blocks near the end of the document so elements on the page that the script interacts are loaded before the script runs.<%@ Reference Control="MyControl.ascx" %>

Q. Explain the differences between Server-side and Client-side code?

A. Server-side code executes on the server. Client-side code executes in the context of the clients' browser.

Q. What are some ways to manage state in an ASP.Net application?

A. Session objects, Application objects, ViewState, cookies, hidden form fields.

Q. What does the "EnableViewState" property do? Why would I want it on or off?

A. It allows page objects to save their state in a Base64 encoded string in the page HTML. One should only have it enabled when needed because it adds to the page size and can get fairly large for complex pages with many controls. (It takes longer to download the page).

Q. What is the difference between Server.Transfer and Response.Redirect ? Why would I choose one over the other?
A. Server.Transfer transfers excution directly to another page. Response.Redirect sends a response to the client and directs the client (the browser) to load the new page (it causes a roundtrip). If you don't need to execute code on the client, Transfer is more efficient.

Q. How can I maintain Session state in a Web Farm or Web Garden?
A. Use a State Server or SQL Server to store the session state.

Q. What base class do all Web Forms inherit from?
A. The Page class.

Q. What does WSDL stand for? What does it do?
A. (Web Services Description Language). It is a way to describe services and how they should be bound to specific network addresses. WSDL has three parts: Definitions, Operations, Service bindings

C# Questions

Q. Can you explain what inheritance is and an example of when you might use it?
A. Inheritance allows us to extend the functionality of a base class. It is an "Is a" type of relationship rather than a "Uses" type of relationship (a dalmation IS A dog which IS A canine which IS A mammal - dalmations inherist from dog which inherits from canine which inherits from mammal). All child classes retain the properties and methods of their parent classes but may override them. When you want to inherit (use the functionality of) another class. Base Class Employee. A Manager class could be derived from the Employee base class.

Q. Does C# support multiple-inheritance?
A. No, use interfaces instead.

Q. Can you prevent your class from being inherited by another class?
A. Yes. The keyword “sealed” will prevent the class from being inherited.

Q. What does the keyword “virtual” declare for a method or property?
A. The method or property can be overridden.

Q. What's the top .NET class that everything is derived from?
A. System.Object.

Q. What does it mean that a String is immutable?
A. Strings cannot be altered. When you alter a string (by adding to it for example), you are actually creating a new string.

Q. If I have to alter a string many times, such as multiple concatenations, what class should I use?
A. StringBuilder. It is not immutable and is very efficient.

Q. In a Try - Catch - Finally block, will the finally block execute if an exception has not occurred? If an Exception has occurred?
A. Yes and yes.

Q. Whats MSIL, and why should developers need an appreciation of it, if at all?
A. MSIL is the Microsoft Intermediate Language. All .NET compatible languages will get converted to MSIL.

Q. Explain the three tier or n-Tier model.
A. Presentation (UI), business (logic and underlying code) and data (from storage or other sources).

Q. What is SOA?
A. Service Oriented Architecture. In SOA you create an abstract layer that your applications use to access various "services" and can aggregate the services. These services could be databases, web services, message queues or other sources. The Service Layer provides a way to access these services that the applications do not need to know how the access is done. For example, to get a full customer record, I might need to get data from a SGL Server database, a web service and a message queue. The Service layer hides this from the calling application. All the application knows is that it asked for a full customer record. It doesn't know what system or systems it came from or how it was retrieved.

Q. What is the role of the DataReader class in ADO.NET connections?
A. It returns a forward-only, read-only view of data from the data source when the command is executed.

Q. Is XML case-sensitive?
A. Yes.

Q. What is the CLR?
A. Common Language Runtime

Q. Can you explain some differences between an ADO.NET Dataset and an ADO Recordset? (Or describe some features of a Dataset).
A. A DataSet can represent an entire relational database in memory, complete with tables, relations, and views. A DataSet is designed to work without any continuing connection to the original data source. Data in a DataSet is bulk-loaded, rather than being loaded on demand.
There's no concept of cursor types in a DataSet. DataSets have no current record pointer You can use For Each loops to move through the data. You can store many edits in a DataSet, and write them to the original data source in a single operation. Though the DataSet is universal, ther objects in ADO.NET come in different versions for different data sources

Q. Name some of the Microsoft Application Blocks. Have you used any? Which ones?
A. Examples:
Exception Management
Logging
Data Access
User Interface
Caching Application Block for .NET
Asynchronous Invocation Application Block for .NET
Configuration Management Application Block for .NET
(there are others) We use Exception and Data Access

Q. Main differences between ASP and ASP.NET.
A. asp contains scrips which are not compiled
where as in asp.net the code is compiled

Q. What is the base class of Button control?
A. system.object

Q. Some of the languages that are supported by .NET
A. 1. Visual Basic.NET
2. Visual C#
3. Visual C++

Q. What is IIS? Have you used it?
A. IIS - Internet Information Server
IIS is used to access the ASP.Net web applications

Q. Main difference between ASP and ASP.NET
A. ASP code will be interpretted
whereas the code written in ASP.NET will be compiled
Q. ADO.NET features
A. 1. Data will be retrieved through Datasets
2. Scalability

Q. Explain assemblies
A. Assemblies are similar to dll files. Both has the reusable pieces of code in the form of classes/ functions. Dll needs to be registered but assemblies have its own metadata.

Q. Explain Assemblies?
A. Assembly is a single deployable unit that contains information about the implementation of classes, structures and interfaces. it also stores the information about itself called metadata and includes name and verison of the assembly, security information, information about the dependencies and the list of files that constitute the assembly.
Assembly also contains namespaces. In the .Net Framework, applications are deployed in the

form of assemblies.

Q. How many types of exception handlers are there in .NET?
A. 1. Unstructured Exception Handling
2. Strutured Exception Handling

Q. ADO.NET features
A. 1. Disconnected Data Architecture
2. Data cached in Datasets
3. Data transfer in XML format
4. Interaction with the database is done through data commands

Q. What is the base class of Button control?
A. Listing from visual studio .net > Button Class
System.Object
System.MarshalByRefObject
System.ComponentModel.Component
System.Windows.Forms.Control
System.Windows.Forms.ButtonBase
System.Windows.Forms.Button

Q. How many types of exception handlers are there in .NET?
A. The exception information table represents four types of exception handlers for protected

blocks:
A finally handler that executes whenever the block exits, whether that occurs by normal control flow or by an unhandled exception.
A fault handler that must execute if an exception occurs, but does not execute on of normal control flow.

A type-filtered handler that handles any exception of a specified class or any of its derived classes.

A user-filtered handler that runs user-specified code to determine whether the exception should be handled by the associated handler or should be passed to the next protected block.

Q. Difference between Panel and GroupBox classes?
A. Panel is scrollable

Q. Difference between Panel and GroupBox classes?
A. Panel & Group box both can used as container for other controls like radio buttons & check box. The difference in panel & group box are Panel

1) In case of panel captions cannot be displayed
2) Can have scroll bars.

Group box

1) Captions can be displayed.
2) Cannot have a scroll bar

Q. What are the advantages and drawbacks of using ADO.NET?
A.

Pros

= = = =

ADO.NET is rich with plenty of features that are bound to impress even the most skeptical of programmers. If this weren't the case, Microsoft wouldn't even be able to get anyone to use the Beta. What we've done here is come up with a short list of some of the more outstanding benefits to using the ADO.NET architecture and the System.Data namespace.

* Performance – there is no doubt that ADO.NET is extremely fast. The actual figures vary depending on who performed the test and which benchmark was being used, but ADO.NET performs much, much faster at the same tasks than its predecessor, ADO. Some of the reasons why ADO.NET is faster than ADO are discussed in the ADO versus ADO.NET section later in this chapter.

* Optimized SQL Provider – in addition to performing well under general circumstances, ADO.NET includes a SQL Server Data Provider that is highly optimized for interaction with SQL Server.

It uses SQL Server's own TDS (Tabular Data Stream) format for exchanging information. Without question, your SQL Server 7 and above data access operations will run blazingly fast utilizing this optimized Data Provider.

* XML Support (and Reliance) – everything you do in ADO.NET at some point will boil down to the use of XML. In fact, many of the classes in

ADO.NET, such as the DataSet, are so intertwined with XML that they simply cannot exist or function without utilizing the technology. You'll see later when we compare and contrast the "old" and the "new" why the reliance on XML for internal storage provides many, many advantages, both to the framework and to the programmer utilizing the class library.

* Disconnected Operation Model – the core ADO.NET class, the DataSet, operates in an entirely disconnected fashion. This may be new to some programmers, but it is a remarkably efficient and scalable architecture. Because the disconnected model allows for the DataSet class to beunaware of the origin of its data, an unlimited number of supported data sources can be plugged into code without any hassle in the future.

* Rich Object Model – the entire ADO.NET architecture is built on a hierarchy of class inheritance and interface implementation. Once you start looking for things you need within this namespace, you'll find that the logical inheritance of features and base class support makes the entire system extremely easy to use, and very customizable to suit your own needs. It is just another example of how everything in the .NET framework is pushing toward a trend of strong application design and strong OOP implementations.

Cons
====
Hard as it may be to believe, there are a couple of drawbacks or disadvantages to using the ADO.NET architecture. I'm sure others can find many more faults than we list here, but we decided to stick with a short list of some of the more obvious and important shortcomings of the technology.

* Managed-Only Access – for a few obvious reasons, and some far more technical, you cannot utilize the ADO.NET architecture from anything but managed code. This means that there is no COM interoperability allowed for ADO.NET. Therefore, in order to take advantage of the advanced SQL Server Data Provider and any other feature like DataSets, XML internal data storage, etc, your code must be running under the CLR.

* Only Three Managed Data Providers (so far) – unfortunately, if you need to access any data that requires a driver that cannot be used through either an OLEDB provider or the SQL Server

Data Provider, then you may be out of luck. However, the good news is that the OLEDB provider for ODBC is available for download from Microsoft. At that point the down-side becomes one of performance, in which you are invoking multiple layers of abstraction as well as crossing the COM InterOp gap, incurring some initial overhead as well.

* Learning Curve – despite the misleading name, ADO.NET is not simply a new version of ADO, nor should it even be considered a direct successor.

ADO.NET should be thought of more as the data access class library for use with the .NET framework. The difficulty in learning to use ADO.NET to its fullest is that a lot of it does seem familiar. It is this that causes some common pitfalls. Programmers need to learn that even though some syntax may appear the same, there is actually a considerable amount of difference in the internal workings of many classes. For example (this will be discussed in far more detail later), an ADO.NET DataSet is nothing at all like a disconnected ADO RecordSet. Some may consider a learning curve a drawback, but I consider learning curves more like scheduling issues. There's a learning curve in learning anything new; it's just up to you to schedule that curve into your time so that you can learn the new technology at a pace that fits your schedule

Q. What are assemblies ?
A. An assembly is a single deployable unit that contains all the information about the

implementation of :
- classes
- structures and
- interfaces

An assembly stores all the information about itself. This information is called METADATA and include the name and the verison number of the assembly, security information, information about the dependencies and a lost of files that constitute the assembly. All the application developed using the .NET framework are made up of assemblies.
Namespaces are also stored in assemblies

Q. What is Response object? How is it related to ASP's Response object?
A. Response object allows the server to communicate with the client (browser). It is useful for displaying information to the user (or) redirecting the client.

Eg: Response.Write("Hello World")

Q. Why The JavaScript Validation Not Run on the Asp.Net Button But Run SuccessFully On The HTML Button

A. The Asp.Net Button Is post backed on the server & not yet Submit & when It goes to the server its states is lost So if we r using javascript in our application so we always use the Input Button in the asp Button

Describe the role of *inetinfo.exe, aspnet_isapi.dll* and*aspnet_wp.exe* in the page loading process.
inetinfo.exe is theMicrosoft IIS server running, handling ASP.NET requests among other things.When an ASP.NET request is received (usually a file with .aspx extension), the ISAPI filter aspnet_isapi.dll takes care of it by passing the request tothe actual worker process aspnet_wp.exe.

What's the difference between Response.Write() andResponse.Output.Write()?
Response.Output.Write() allows you to write formatted output.

What methods are fired during the page load?
Init() - when the page is instantiated
Load() - when the page is loaded into server memory
PreRender() - the brief moment before the page is displayed to the user as HTML
Unload() - when page finishes loading

When during the page processing cycle is ViewState available?
After the Init() and before the Page_Load(), or OnLoad() for a control.

What namespace does the Web page belong in the .NET Framework class hierarchy?
System.Web.UI.Page

What's the difference between Codebehind="MyCode.aspx.cs" andSrc="MyCode.aspx.cs"?
CodeBehind is relevant to Visual Studio.NET only.

What's a bubbled event?
When you have a complex control, like DataGrid, writing an event processing routine for each object (cell, button, row, etc.) is quite tedious. The controls can bubble up their eventhandlers, allowing the main DataGrid event handler to take care of its constituents.

Suppose you want a certain ASP.NET function executed on MouseOver for a certain button. Where do you add an event handler?
Add an OnMouseOver attribute to the button. Example:
btnSubmit.Attributes.Add("onmouseover","someClientCodeHere();");

What data types do the RangeValidator control support?
Integer, String, and Date

Explain the differences between Server-side and Client-side code?
Server-side code executes on the server. Client-side code executes in the client's browser.

What type of code (server or client) is found in a Code-Behind class?
The answer is server-side code since code-behind is executed on the server. However, during the code-behind's execution on the server, it can render client-side code such as JavaScript to be processed in the clients browser. But just to be clear, code-behind executes on the server, thus making it server-side code

Should user input data validation occur server-side or client-side? Why?
All user input data validation should occur on the server at a minimum. Additionally, client-side validation can be performed where deemed appropriate and feasable to provide a richer, more responsive experience for the user.

What is the difference between Server.Transfer and Response.Redirect? Why would I choose one over the other?
Server.Transfer transfers page processing from one page directly to the next page without making a round-trip back to the client's browser. This provides a faster response with a little less overhead on the server. Server.Transfer does not update the clients url history list or current url. Response.Redirect is used to redirect the user's browser to another page or site. This performas a trip back to the client where the client's browser is redirected to the new page. The user's browser history list is updated to reflect the new address.

Can you explain the difference between an ADO.NET Dataset and an ADO Recordset?
Valid answers are:
· A DataSet can represent an entire relational database in memory, complete with tables, relations, and views.
· A DataSet is designed to work without any continuing connection to the original data source.
· Data in a DataSet is bulk-loaded, rather than being loaded on demand.
· There's no concept of cursor types in a DataSet.
· DataSets have no current record pointer You can use For Each loops to move through the data.
· You can store many edits in a DataSet, and write them to the original data source in a single operation.
· Though the DataSet is universal, other objects in ADO.NET come in different versions for different data sources.

What is the Global.asax used for?
The Global.asax (including the Global.asax.cs file) is used to implement application and session level events.

What are the Application_Start and Session_Start subroutines used for?
This is where you can set the specific variables for the Application and Session objects.

Can you explain what inheritance is and an example of when you might use it?
When you want to inherit (use the functionality of) another class. Example: With a base class named Employee, a Manager class could be derived from the Employee base class.

Whats an assembly?
Assemblies are the building blocks of the .NET framework

Describe the difference between inline and code behind.
Inline code written along side the html in a page. Code-behind is code written in a separate file and referenced by the .aspx page

Explain what a diffgram is, and a good use for one?
The DiffGram is one of the two XML formats that you can use to render DataSet object contents to XML. A good use is reading database data to an XML file to be sent to a Web Service

Whats MSIL, and why should my developers need an appreciation of it if at all?
MSIL is the Microsoft Intermediate Language. All .NET compatible languages will get converted to MSIL. MSIL also allows the .NET Framework to JIT compile the assembly on the installed computer

Which method do you invoke on the DataAdapter control to load your generated dataset with data?
The Fill() method

Can you edit data in the Repeater control?
No, it just reads the information from its data source

Which template must you provide, in order to display data in a Repeater control?
ItemTemplate.

How can you provide an alternating color scheme in a Repeater control?
Use the AlternatingItemTemplate

What property must you set, and what method must you call in your code, in order to bind the data from a data source to the Repeater control?
You must set the DataSource property and call the DataBind method

What base class do all Web Forms inherit from?
The Page class.

Name two properties common in every validation control?
ControlToValidate property and Text property.

Which property on a Combo Box do you set with a column name, prior to setting the DataSource, to display data in the combo box?
DataTextField property.

Which control would you use if you needed to make sure the values in two different controls matched?
CompareValidator control.

How many classes can a single .NET DLL contain?
It can contain many classes.

web service.

Q. Which WebForm Validator control would you use if you needed to make sure the values in two different WebForm controls matched?
A. CompareValidator Control

Q. What property must you set, and what method must you call in your code, in order to bind the data from some data source to the Repeater control?
A. You must set the DataSource property and call the DataBind method.

Web Service Questions

1. **What is the transport protocol you use to call a Web service?**
   SOAP (Simple Object Access Protocol) is the preferred protocol.

2. **True or False: A Web service can only be written in .NET?**
   False

3. **What does WSDL stand for?**
   Web Services Description Language.

4. **Where on the Internet would you look for Web services?**
   http://www.uddi.org

5. **True or False: To test a Web service you must create a Windows application or Web application to consume this service?**
   False, the web service comes with a test page and it provides HTTP-GET method to test.

State Management Questions

1. **What is ViewState?**
   ViewState allows the state of objects (serializable) to be stored in a hidden field on the page. ViewState is transported to the client and back to the server, and is not stored on the server or any other external source. ViewState is used the retain the state of server-side objects between postabacks.

2. **What is the lifespan for items stored in ViewState?**
   Item stored in ViewState exist for the life of the current page. This includes postbacks (to the same page).

3. **What does the "EnableViewState" property do? Why would I want it on or off?**
   It allows the page to save the users input on a form across postbacks. It saves the server-side values for a given control into ViewState, which is stored as a hidden value on the page before sending the page to the clients browser. When the page is posted back to the server the server control is recreated with the state stored in viewstate.

4. **What are the different types of Session state management options available with ASP.NET?**
   ASP.NET provides In-Process and Out-of-Process state management. In-Process stores the session in memory on the web server. This requires the a "sticky-server" (or no load-balancing) so that the user is always reconnected to the same web server. Out-of-Process Session state management stores data in an external data source. The external data source may be either a SQL Server or a State Server service. Out-of-Process state management requires that all objects stored in session are serializable.

C# Interview Questions

General Questions

1. **Does C# support multiple-inheritance?**
   No.

2. **Who is a protected class-level variable available to?**
   It is available to any sub-class (a class inheriting this class).

3. **Are private class-level variables inherited?**
   Yes, but they are not accessible. Although they are not visible or accessible via the class interface, they are inherited.

4. **Describe the accessibility modifier "protected internal".**
   It is available to classes that are within the same assembly and derived from the specified base class.

5. **What's the top .NET class that everything is derived from?**
   System.Object.

6. **What does the term immutable mean?**
   The data value may not be changed. Note: The *variable* value may be changed, but the original immutable data value was discarded and a new data value was created in memory.

7. **What's the difference between System.String and System.Text.StringBuilder classes?**
   System.String is immutable. System.StringBuilder was designed with the purpose of having a mutable string where a variety of operations can be performed.

8. **What's the advantage of using System.Text.StringBuilder over System.String?**
   StringBuilder is more efficient in cases where there is a large amount of string manipulation. Strings are immutable, so each time a string is changed, a new instance in memory is created.

9. **Can you store multiple data types in System.Array?**
   No.

10. **What's the difference between the System.Array.CopyTo() and System.Array.Clone()?**
    The first one performs a deep copy of the array, the second one is shallow. A shallow copy of an Array copies only the elements of the **Array**, whether they are reference types or value types, but it does not copy the objects that the references refer to. The references in the new **Array** point to the same objects that the references in the original **Array** point to. In contrast, a deep copy of an **Array** copies the elements and everything directly or indirectly referenced by the elements.

11. **How can you sort the elements of the array in descending order?**
    By calling Sort() and then Reverse() methods.

12. **What's the .NET collection class that allows an element to be accessed using a unique key?**
    HashTable.

13. **What class is underneath the SortedList class?**
A sorted HashTable.

14. **Will the finally block get executed if an exception has not occurred?**
Yes.

15. **What's the C# syntax to catch any possible exception?**
A catch block that catches the exception of type System.Exception. You can also omit the parameter data type in this case and just write catch {}.

16. **Can multiple catch blocks be executed for a single try statement?**
No. Once the proper catch block processed, control is transferred to the finally block (if there are any).

17. **Explain the three services model commonly know as a three-tier application.**
Presentation (UI), Business (logic and underlying code) and Data (from storage or other sources).

Class Questions

1. **What is the syntax to inherit from a class in C#?**
Place a colon and then the name of the base class.
Example: *class MyNewClass : MyBaseClass*

2. **Can you prevent your class from being inherited by another class?**
Yes. The keyword "sealed" will prevent the class from being inherited.

3. **Can you allow a class to be inherited, but prevent the method from being over-ridden?**
Yes. Just leave the class public and make the method sealed.

4. **What's an abstract class?**
A class that cannot be instantiated. An abstract class is a class that must be inherited and have the methods overridden. An abstract class is essentially a blueprint for a class without any implementation.

5. **When do you absolutely have to declare a class as abstract?**
1. When the class itself is inherited from an abstract class, but not all base abstract methods have been overridden.

2. When at least one of the methods in the class is abstract.

6. **What is an interface class?**
   Interfaces, like classes, define a set of properties, methods, and events. But unlike classes, interfaces do not provide implementation. They are implemented by classes, and defined as separate entities from classes.

7. **Why can't you specify the accessibility modifier for methods inside the interface?**
   They all must be public, and are therefore public by default.

8. **Can you inherit multiple interfaces?**
   Yes. .NET does support multiple interfaces.

9. **What happens if you inherit multiple interfaces and they have conflicting method names?**
   It's up to you to implement the method inside your own class, so implementation is left entirely up to you. This might cause a problem on a higher-level scale if similarly named methods from different interfaces expect different data, but as far as compiler cares you're okay.
   To Do: Investigate

10. **What's the difference between an interface and abstract class?**
    In an interface class, all methods are abstract - there is no implementation. In an abstract class some methods can be concrete. In an interface class, no accessibility modifiers are allowed. An abstract class may have accessibility modifiers.

11. **What is the difference between a Struct and a Class?**
    Structs are value-type variables and are thus saved on the stack, additional overhead but faster retrieval. Another difference is that structs **cannot** inherit.

Method and Property Questions

1. **What's the implicit name of the parameter that gets passed into the set method / property of a class?**
   Value. The data type of the value parameter is defined by whatever data type the property is declared as.

2. **What does the keyword "virtual" declare for a method or property?**

The method or property can be overridden.

3. **How is method overriding different from method overloading?**
   When overriding a method, you change the behavior of the method for the derived class. Overloading a method simply involves having another method with the same name within the class.

4. **Can you declare an override method to be static if the original method is not static?**
   No. The signature of the virtual method must remain the same. (Note: Only the keyword virtual is changed to keyword override)

5. **What are the different ways a method can be overloaded?**
   Different parameter data types, different number of parameters, different order of parameters.

6. **If a base class has a number of overloaded constructors, and an inheriting class has a number of overloaded constructors; can you enforce a call from an inherited constructor to a specific base constructor?**
   Yes, just place a colon, and then keyword base (parameter list to invoke the appropriate constructor) in the overloaded constructor definition inside the inherited class.

Events and Delegates

1. **What's a delegate?**
   A delegate object encapsulates a reference to a method.

2. **What's a multicast delegate?**
   A delegate that has multiple handlers assigned to it. Each assigned handler (method) is called.

XML Documentation Questions

1. **Is XML case-sensitive?**
   Yes.

What's the difference between // comments, /* */ comments and /// comments?
Single-line comments, multi-line comments, and XML documentation comments.

3. **How do you generate documentation from the C# file commented properly with a command-line compiler?**
   Compile it with the /doc switch.