**.Net Framework Part 2**

**7. What is mean by JIT?**

Just In Time(JIT) compilers which compile the IL code to native executable code (.exe or .dll) for the specific machine and OS. JIT are slightly different from traditional compilers as they compile the IL to native code only when desired e.g., when a function is called, IL of function's body is converted to native code; just in time of need. If same function is called next time, the CLR uses the same copy of native code without re-compiling. As JIT are aware of processor and OS exactly at runtime, they can optimize the code extremely efficiently resulting in very robust applications.

**8. What are different types of JIT?**

**Pre-JIT**- Pre-JIT compiles complete source code into native code in a single compilation cycle. This is done at the time of deployment of the application.

**Econo-JIT**- Econo-JIT compiles only those functions that are called at runtime. However, these compiled functions are removed when they are not required.

**Normal-JIT**- Normal-JIT compiles only those functions that are called at runtime and they are stored in cache. If same function is called next time, the CLR uses the same copy of compiled code without re-compiling.

**9. What is mean by Assembly?**

* Assemblies are self-describing installation units, consisting of one or more files.
* Assemblies are the deployment units of .Net applications. .Net application consists of one or more assemblies.
* An assembly may also contain references to other assemblies and it include metadata that describes all the types that are defined in the assembly with information about it members-methods, properties, events and fields.
* One assembly could be a single Dll or exe that includes metadata, or it can be made of different files e.g resource files, modules and an exe.
* Assembly manifests is a part of the metadata, it describes the assembly with all the information that's needed to reference it and lists all its dependencies.

**10. What are the features of Assembly?**

* Assemblies are *self-describing*, it includes metadata that describes the assembly. It does not required to register key as like COM component.
* Version dependencies are recorded inside an assembly manifest. The version of the referenced assembly that will be used can be configured by the developer and the system administrator.
* Two different version of same assembly can be used inside single process.

**11. What are different type's assemblies?**

**Private assembly**- Private assembly is used within your application and it is installed at the same time as the application itself. It will be located in the same directory as the application or subdirectories thereof.

**Shared assembly**- Shared assemblies are used by several application. Shared assembly must have version number and unique name and it is usually installed in GAC (Global assembly catch). It reduces the need for disk space and memory space.

**12. What are parts of assembly manifests?**

* Identity - Name, version, culture and public key
* A list of files - Files belonging to the assembly, it can have one or more files.
* Lists of referenced assemblies - all assemblies referenced by the assembly are documented inside the manifest.
* Set of permission requests- these are the permission needed to run the assembly.
* Exported types - It describes the structures, class with properties, method and events. It is used to create instance of the class.