[HTML 5 11](#_Toc521765445)

[New Elements in HTML5 11](#_Toc521765446)

[HTML5 offers new elements for better document structure: 11](#_Toc521765447)

[New Semantic Elements in HTML5 12](#_Toc521765448)

[JavaScript 13](#_Toc521765449)

[jQuery: 13](#_Toc521765450)

[Framework 13](#_Toc521765451)

[What is concept of Boxing and Unboxing ? 13](#_Toc521765452)

[What is the difference between System exceptions and Application exceptions? 13](#_Toc521765453)

[Access Modifier 14](#_Toc521765454)

[**Anonymous Type vs Dynamic Type** 14](#_Toc521765455)

[ASP.NET 14](#_Toc521765456)

[**What is the difference between Server.Transfer and response.Redirect ?** 14](#_Toc521765457)

[**Describe state management in ASP.NET.** 14](#_Toc521765458)

[**Explain client side state management system.** 15](#_Toc521765459)

[**Explain server side state management system.** 15](#_Toc521765460)

[What is Session object? Describe in detail 15](#_Toc521765461)

[Differentiate between a HyperLink control and a LinkButton control. 16](#_Toc521765462)

[What are the various types of validation controls provided by ASP.NET? 16](#_Toc521765463)

[What is caching? 17](#_Toc521765464)

[Explain the use of duration attribute of @OutputCache page directive. 17](#_Toc521765465)

[Describe the application event handlers in ASP.NET 17](#_Toc521765466)

[What are the Web Form Events available in ASP.NET? 17](#_Toc521765467)

[What are the options in ASP.NET to maintain state? 18](#_Toc521765468)

[What is delegate? 18](#_Toc521765469)

[How you can access the Properties and Controls of Master Pages from content pages? 18](#_Toc521765470)

[What is reflection? 19](#_Toc521765471)

[What is AutoPostback? 19](#_Toc521765472)

[How to turn off cookies for a page? 19](#_Toc521765473)

[Response.Redirect vs Server.Transfer 19](#_Toc521765474)

[List down the sequence of methods called during the page load. 20](#_Toc521765475)

[Define authentication and authorization. 20](#_Toc521765476)

[ASP.NET Performance 20](#_Toc521765477)

[Json 21](#_Toc521765478)

[**Why Json?** 21](#_Toc521765479)

[LINQ 21](#_Toc521765480)

[LINQ Single vs SingleOrDefault vs First vs FirstOrDefault 21](#_Toc521765481)

[Entity Framework 22](#_Toc521765482)

[Advantage 22](#_Toc521765483)

[Entity Framework 4 vs NHibernate 22](#_Toc521765484)

[Using Code First model, how can I mark a field/property as the primary key if it does not follow the code first convention? 23](#_Toc521765485)

[When you have a annotate a property as Primary key in a table, how do you enable foreign key relationship from another table? 24](#_Toc521765486)

[How do you mark a property as required? For example, For a Project, the Name is a required field. 24](#_Toc521765487)

[How do you enforce a field to have a minimum and maximum number of characters? For example, the Description on a Project should be a minimu of 10 and a maximum of 500? 24](#_Toc521765488)

[Define a property in project class named ProjectCode that is not mapped to the database. ProjectCode is internally calculated as a combination of project ID and Title. 25](#_Toc521765489)

[Define a property in project class named ProjectCode that is not mapped to the database. ProjectCode is internally calculated as a combination of project ID and Title. 25](#_Toc521765490)

[How can you tell EF to have a different table or column name than that defined for the class? 26](#_Toc521765491)

[When two tables have multiple relationships (for example, a task is created by employee 1 and updated by employee 2), who do you indicate which relationships go with which property? 27](#_Toc521765492)

[Lezy load 27](#_Toc521765493)

[Eagerly loading related entities 28](#_Toc521765494)

[OOPS 29](#_Toc521765495)

[Abstraction 29](#_Toc521765496)

[Association 29](#_Toc521765497)

[Aggregation 29](#_Toc521765498)

[Inheritance 29](#_Toc521765499)

[Polymorphism 29](#_Toc521765500)

[What is difference between abstract classes and interfaces? 29](#_Toc521765501)

[Generic Classes 29](#_Toc521765502)

[Generic Data Type 30](#_Toc521765503)

[MVC 30](#_Toc521765504)

[Explain MVC application life cycle? 30](#_Toc521765505)

[HTTPGet 31](#_Toc521765506)

[HTTPPost 31](#_Toc521765507)

[What is ASP.NET MVC? 32](#_Toc521765508)

[Difference between ASP.NET WebForms and ASP.NET MVC 32](#_Toc521765509)

[Rest Web Service 35](#_Toc521765510)

[What is a RESTful Web Service? 35](#_Toc521765511)

[WCF 36](#_Toc521765512)

[Binding 36](#_Toc521765513)

[Contract 36](#_Toc521765514)

[Service Contract 37](#_Toc521765515)

[Data Contract 37](#_Toc521765516)

[What is DataContractSerializer and How its different from XmlSerializer? 37](#_Toc521765517)

[Please explain briefly different Instance Modes in WCF? 38](#_Toc521765518)

[What Message Exchange Patterns (MEPs) supported by WCF? Explain each of them briefly. 38](#_Toc521765519)

[What are contracts in WCF? In WCF, all services expose contracts. The contract is a platform-neutral and standard way of describing what the service does. WCF defines four types of contracts. 38](#_Toc521765520)

[What is WCF throttling? 40](#_Toc521765521)

[Please explain about authorization options supported in WCF? 40](#_Toc521765522)

[What is Reliable Messaging in WCF? 40](#_Toc521765523)

[What are Reliable Sessions in WCF? 41](#_Toc521765524)

[WCF Security 41](#_Toc521765525)

[**No transfer security mode** 41](#_Toc521765526)

[Transport security mode 41](#_Toc521765527)

[Message security mode 42](#_Toc521765528)

[Mixed transfer security mode 42](#_Toc521765529)

[Both security modes 42](#_Toc521765530)

[SQL SERVER 42](#_Toc521765531)

[Normalization 42](#_Toc521765532)

[What is RDBMS? 43](#_Toc521765533)

[What is the difference between a HAVING CLAUSE and a WHERE CLAUSE? 43](#_Toc521765534)

[What is the difference between UNION and UNION ALL SQL syntax ? 43](#_Toc521765535)

[Efficiently convert rows to columns in sql server 43](#_Toc521765536)

[Database ACID (Atomicity, Consistency, Isolation, Durability) Properties 44](#_Toc521765537)

[**What is difference between DELETE and TRUNCATE commands?** 45](#_Toc521765538)

[What are the Different Normalization Forms? 45](#_Toc521765539)

[What is a Stored Procedure? 45](#_Toc521765540)

[What is a Trigger? 46](#_Toc521765541)

[What are the Different Types of Triggers? 46](#_Toc521765542)

[What is a View? 46](#_Toc521765543)

[What is an Index? 46](#_Toc521765544)

[What is a Linked Server? 46](#_Toc521765545)

[What is a Cursor? 46](#_Toc521765546)

[What is Collation? 46](#_Toc521765547)

[What is the Difference between a Function and a Stored Procedure? 46](#_Toc521765548)

[What is subquery? Explain the Properties of a Subquery? 47](#_Toc521765549)

[What are Different Types of Join? 47](#_Toc521765550)

[What are Primary Keys and Foreign Keys? 47](#_Toc521765551)

[What is User-defined Functions? What are the types of User-defined Functions that can be created? 47](#_Toc521765552)

[What is an Identity? 47](#_Toc521765553)

[What is DataWarehousing? 47](#_Toc521765554)

[What languages BI uses to achieve the goal? 47](#_Toc521765555)

[What is Standby Servers? Explain Types of Standby Servers. 47](#_Toc521765556)

[What is Dirty Read? 47](#_Toc521765557)

[Why can’t I use Outer Join in an Indexed View? 47](#_Toc521765558)

[What is the Correct Order of the Logical Query Processing Phases? 47](#_Toc521765559)

[What are Different Types of Locks? 47](#_Toc521765560)

[What are Pessimistic Lock and Optimistic Lock? 47](#_Toc521765561)

[When is the use of UPDATE\_STATISTICS command? 47](#_Toc521765562)

[What is the Difference between a HAVING clause and a WHERE clause? 47](#_Toc521765563)

[What is Connection Pooling and why it is Used? 47](#_Toc521765564)

[What are the Properties and Different Types of Sub-Queries? 47](#_Toc521765565)

[What are the Authentication Modes in SQL Server? How can it be Changed? 47](#_Toc521765566)

[Which Command using Query Analyzer will give you the Version of SQL Server and Operating System? 47](#_Toc521765567)

[What is an SQL Server Agent? 47](#_Toc521765568)

[Can a Stored Procedure call itself or a Recursive Stored Procedure? How many levels of SP nesting is possible? 47](#_Toc521765569)

[What is Log Shipping? 47](#_Toc521765570)

[Name 3 ways to get an Accurate Count of the Number of Records in a Table? 47](#_Toc521765571)

[What does it mean to have QUOTED\_IDENTIFIER ON? What are the Implications of having it OFF? 47](#_Toc521765572)

[What is the Difference between a Local and a Global Temporary Table? 47](#_Toc521765573)

[What is the STUFF Function and How Does it Differ from the REPLACE Function? 47](#_Toc521765574)

[What is PRIMARY KEY? 47](#_Toc521765575)

[What is UNIQUE KEY Constraint? 47](#_Toc521765576)

[What is FOREIGN KEY? 47](#_Toc521765577)

[What is CHECK Constraint? 47](#_Toc521765578)

[What is NOT NULL Constraint? 48](#_Toc521765579)

[What is the difference between UNION and UNION ALL? 48](#_Toc521765580)

[What is B-Tree? 48](#_Toc521765581)

[How to get @@ERROR and @@ROWCOUNT at the Same Time? 48](#_Toc521765582)

[What is a Scheduled Job or What is a Scheduled Task? 48](#_Toc521765583)

[What are the Advantages of Using Stored Procedures? 48](#_Toc521765584)

[What is a Table Called, if it has neither Cluster nor Non-cluster Index? What is it Used for? 48](#_Toc521765585)

[Can SQL Servers Linked to other Servers like Oracle? 48](#_Toc521765586)

[What is BCP? When is it Used? 48](#_Toc521765587)

[What Command do we Use to Rename a db, a Table and a Column? 48](#_Toc521765588)

[What are sp\_configure Commands and SET Commands? 48](#_Toc521765589)

[How to Implement One-to-One, One-to-Many and Many-to-Many Relationships while Designing Tables? 48](#_Toc521765590)

[What is Difference between Commit and Rollback when Used in Transactions? 48](#_Toc521765591)

[What is an Execution Plan? When would you Use it? How would you View the Execution Plan? 48](#_Toc521765592)

[What is Difference between Table Aliases and Column Aliases? Do they Affect Performance? 48](#_Toc521765593)

[What is the difference between CHAR and VARCHAR Datatypes? 48](#_Toc521765594)

[What is the Difference between VARCHAR and VARCHAR(MAX) Datatypes? 48](#_Toc521765595)

[What is the Difference between VARCHAR and NVARCHAR datatypes? 48](#_Toc521765596)

[Which are the Important Points to Note when Multilanguage Data is Stored in a Table? 48](#_Toc521765597)

[How to Optimize Stored Procedure Optimization? 48](#_Toc521765598)

[What is SQL Injection? How to Protect Against SQL Injection Attack? 48](#_Toc521765599)

[How to Find Out the List Schema Name and Table Name for the Database? 48](#_Toc521765600)

[What is CHECKPOINT Process in the SQL Server? 48](#_Toc521765601)

[How does Using a Separate Hard Drive for Several Database Objects Improves Performance Right Away? 48](#_Toc521765602)

[How to Find the List of Fixed Hard Drive and Free Space on Server? 48](#_Toc521765603)

[Why can there be only one Clustered Index and not more than one? 48](#_Toc521765604)

[What is Difference between Line Feed (\n) and Carriage Return (\r)? 48](#_Toc521765605)

[Is It Possible to have Clustered Index on Separate Drive From Original Table Location? 48](#_Toc521765606)

[What is a Hint? 48](#_Toc521765607)

[How to Delete Duplicate Rows? 48](#_Toc521765608)

[Why the Trigger Fires Multiple Times in Single Login? 49](#_Toc521765609)

[What is Aggregate Functions? 49](#_Toc521765610)

[What is Use of @@ SPID in SQL Server? 49](#_Toc521765611)

[What is the Difference between Index Seek vs. Index Scan? 49](#_Toc521765612)

[What is the Maximum Size per Database for SQL Server Express? 49](#_Toc521765613)

[How do We Know if Any Query is Retrieving a Large Amount of Data or very little data? 49](#_Toc521765614)

[What is the Difference between GRANT and WITH GRANT while Giving Permissions to the User? 49](#_Toc521765615)

[How to Create Primary Key with Specific Name while Creating a Table? 49](#_Toc521765616)

[What is T-SQL Script to Take Database Offline – Take Database Online 49](#_Toc521765617)

[How to Enable/Disable Indexes? 49](#_Toc521765618)

[Can we Insert Data if Clustered Index is Disabled? 49](#_Toc521765619)

[How to Recompile Stored Procedure at Run Time? 49](#_Toc521765620)

[Is there any Performance Difference between IF EXISTS (Select null from table) and IF EXISTS (Select 1 from table)? 49](#_Toc521765621)

[What is Difference in Performance between INSERT TOP (N) INTO Table and Using Top with INSERT? 49](#_Toc521765622)

[Does the Order of Columns in UPDATE statements Matter? 49](#_Toc521765623)

[What are the basic functions for master, msdb, model, tempdb and resource databases? 49](#_Toc521765624)

[What is the Maximum Number of Index per Table? 49](#_Toc521765625)

[Explain Few of the New Features of SQL Server 2008 Management Studio 49](#_Toc521765626)

[Explain IntelliSense for Query Editing 49](#_Toc521765627)

[Explain MultiServer Query 49](#_Toc521765628)

[Explain Query Editor Regions 49](#_Toc521765629)

[Explain Object Explorer Enhancements 49](#_Toc521765630)

[Explain Activity Monitors 49](#_Toc521765631)

[What is Service Broker? 49](#_Toc521765632)

[Where are SQL server Usernames and Passwords Stored in the SQL server? 49](#_Toc521765633)

[What is Policy Management? 49](#_Toc521765634)

[What is Database Mirroring? 49](#_Toc521765635)

[What are Sparse Columns? 49](#_Toc521765636)

[What does TOP Operator Do? 49](#_Toc521765637)

[What is CTE? 49](#_Toc521765638)

[What is MERGE Statement? 50](#_Toc521765639)

[What is Filtered Index? 50](#_Toc521765640)

[Which are the New Data Types Introduced in SQL SERVER 2008? 50](#_Toc521765641)

[What are the Advantages of Using CTE? 50](#_Toc521765642)

[How can we Rewrite Sub-Queries into Simple Select Statements or with Joins? 50](#_Toc521765643)

[What is CLR? 50](#_Toc521765644)

[What are Synonyms? 50](#_Toc521765645)

[What is LINQ? 50](#_Toc521765646)

[What are Isolation Levels? 50](#_Toc521765647)

[What is Use of EXCEPT Clause? 50](#_Toc521765648)

[What is XPath? 50](#_Toc521765649)

[What is NOLOCK? 50](#_Toc521765650)

[What is the Difference between Update Lock and Exclusive Lock? 50](#_Toc521765651)

[How to Copy Data from One Table to Another Table? 50](#_Toc521765652)

[What is Catalog Views? 50](#_Toc521765653)

[What is PIVOT and UNPIVOT? 50](#_Toc521765654)

[What is a Filestream? 50](#_Toc521765655)

[What is SQLCMD? 50](#_Toc521765656)

[What do you mean by TABLESAMPLE? 50](#_Toc521765657)

[What is ROW\_NUMBER()? 50](#_Toc521765658)

[What are Ranking Functions? 50](#_Toc521765659)

[What is Change Data Capture (CDC) in SQL Server 2008? 50](#_Toc521765660)

[How can I Track the Changes or Identify the Latest Insert-Update-Delete from a Table? 50](#_Toc521765661)

[What is the CPU Pressure? 50](#_Toc521765662)

[How can I Get Data from a Database on Another Server? 50](#_Toc521765663)

[What is the Bookmark Lookup and RID Lookup? 50](#_Toc521765664)

[What is Difference between ROLLBACK IMMEDIATE and WITH NO\_WAIT during ALTER DATABASE? 50](#_Toc521765665)

[What is Difference between GETDATE and SYSDATETIME in SQL Server 2008? 50](#_Toc521765666)

[How can I Check that whether Automatic Statistic Update is Enabled or not? 50](#_Toc521765667)

[How to Find Index Size for Each Index on Table? 50](#_Toc521765668)

[What is the Difference between Seek Predicate and Predicate? 51](#_Toc521765669)

[What are Basics of Policy Management? 51](#_Toc521765670)

[What are the Advantages of Policy Management? 51](#_Toc521765671)

[Window Azure 51](#_Toc521765672)

[Components of AZURE 51](#_Toc521765673)

[Table Storage Batch Operation 51](#_Toc521765674)

[Windows Azure Portal 52](#_Toc521765675)

[How does the Azure fabric communicate with computing and storage services? 52](#_Toc521765676)

[**What is Blob Storage?** 52](#_Toc521765677)

[***Blob Snapshots*** 53](#_Toc521765678)

[**What is the Table Service?** 53](#_Toc521765679)

[What is Windows Azure? 54](#_Toc521765680)

[**What is Windows Azure Platform?** 54](#_Toc521765681)

[Compute: It provides a computation environment with Web Role, Worker Role, and VM Role. 54](#_Toc521765682)

[Storage: It focuses on providing scalable storage (Blobs, non-relational Tables, and Queues) for large-scale needs. Relational Database functionality is offered through SQL Azure, which is a scalable version of SQL Server that runs on the Azure platform. 54](#_Toc521765683)

[Fabric: Fabric (Windows Azure Fabric) makes up the physical underpinnings of the Windows Azure platform as the network of interconnected nodes consisting of servers, high-speed connections, and switches. Conceptually, the repetitive pattern of nodes and connections suggests a woven or fabric-like nature. Compute and Storage components are part of the Fabric. 54](#_Toc521765684)

[**What is the downtime for applications in case of any patching? Ans.** Windows Azure will have replicas for each and every application and provide zero downtime in case of any security patching. 54](#_Toc521765685)

[**How many copies of data are maintained in Windows Azure? Ans.** Windows Azure provides you with three copies of data. This makes your application running on very reliable data. 54](#_Toc521765686)

[**What is queue storage in the Windows Azure? Ans.** Queue storage gives you capability of sending the small data just as messages. Queue storage basically helps informing the task to the worker threads. 55](#_Toc521765687)

[**What is Windows Azure AppFabric? What are the services it provides? Ans.** Windows Azure AppFabric is a part of the Microsoft Windows Azure Platform. Positioned by Microsoft as middleware, it provides several cloud computing services: 55](#_Toc521765688)

[**Access Control Service:** Provides an easy way for web applications and web services to identify users, and grant access control to the applications and services. 55](#_Toc521765689)

[**AppFabric Applications:** Provides tools and APIs for developing and hosting a composite application expressly for deployment through the service. 55](#_Toc521765690)

[**Service Bus:** Provides secure connections between distributed and disconnected applications in the cloud. 55](#_Toc521765691)

[**Caching:** Provides a distributed, in-memory, application cache service for Windows Azure and SQL Azure applications. 55](#_Toc521765692)

[**Integration:** Leverages several capabilities of Microsoft BizTalk Server such as transforming messages from one format to another, or providing a queue for message passing. 55](#_Toc521765693)

[**What are the main uses of Windows Azure? Ans.** 1. Run enterprise workloads in the cloud 2. Build, modify, and distribute scalable applications with minimal on-premises resources 3. Perform large-volume storage, batch processing, intense or large-volume computations 4. Create, test, debug, and distribute Web services quickly and inexpensively 55](#_Toc521765694)

[**What are benefits of Windows Azure?** Bring your ideas to market faster and pay as you go 2. Reduce costs of building and extending on-premises resources 3. Reduce the effort and costs of IT management 4. Respond quickly to changes in your business and customer needs 5. Choose an on-premises or off-premises deployment model that best suits your needs. 6. Scale your IT resources up and down based on your needs. 7. Consume computing resources ONLY when the needs arise. 8. Focus less energy on managing operational resources and constraints. 9. Remove the need to manage hardware 10. Use your existing development skills to build cloud applications 11. Consistent development and management experience across on-premises and the cloud. 55](#_Toc521765695)

[**What is worker role?**  A role that provides a general-purpose environment for running application workloads. Typically, background processing for web applications running on web roles can be offloaded to instances of the worker role. 56](#_Toc521765696)

[**What is web role?** Ans. A role that provides an environment for running web sites or applications as supported by Internet Information Services (IIS) 7.0. 56](#_Toc521765697)

[**What do you mean by VM role?** Ans. A role that provides a customer-customizable virtual machine in which to run the customer’s hosted service. 56](#_Toc521765698)

[**What is Virtual IP(VIP)?** An IP address that is not connected to a specific computer or network interface card. Service deployments are assigned a VIP for receiving network traffic which is redirected to a physical device in the Windows Azure fabric. 56](#_Toc521765699)

[**What is use of activation token?**  A unique identifier that activates a Windows Azure Connect endpoint and is linked to the customer’s subscription. The token can be reset by the customer. 56](#_Toc521765700)

[**What do you mean by affinity group?** Ans. A geographical grouping of a customer’s hosted service deployments and storage accounts within Windows Azure. An affinity group can improve service performance by locating compute and data storage workloads in the same data center or near the target user audience. 56](#_Toc521765701)

[**What is cloud?**  A collection of servers, networks, and gateways that provide a computing environment that is delivered over the Internet as a service on a pay-per-use basis. 56](#_Toc521765702)

[What do you mean by Compute Emulator and Compute hours? 56](#_Toc521765703)

[**What is a container?** Ans. It is a user-defined set of blobs within a storage account. A container resource has no associated content, only properties and metadata. 57](#_Toc521765704)

[**What is cspack?** Ans. It is a command-line tool that generates a service package file (.cspkg) and prepares an application for deployment, either to Windows Azure or to the compute emulator. 57](#_Toc521765705)

[**What is csrun?**  It is a command-line tool that deploys a packaged application to the Windows Azure compute emulator and manages the running service. 57](#_Toc521765706)

[**What is guest OS?**  It is the operating system that runs on the virtual machine that hosts an instance of a role. 57](#_Toc521765707)

[**What is input endpoint?**  It is the IP and port on which a role instance receives inbound traffic. Each type of role has restrictions on the number and type of input endpoints that can be defined. A web role can have no more than one HTTP endpoint and one HTTPS endpoint. A work role can have no more than five HTTP, HTTPS, or TCP endpoints (any combination). A VM role can have only one HTTP, HTTPS, or TCP endpoint. 57](#_Toc521765708)

[**What do you mean by page blob?**  It is a collection of pages in a blob. A page is a range of data that is identified by its offset from the start of the blob. 57](#_Toc521765709)

[***What is a Fabric Agent?*** 57](#_Toc521765710)

[**What is Content Delivery Network (CDN)?** An add-on feature to Windows Azure subscription to cache Windows Azure BLOBs and the static content output of Compute instances at Microsoft’s caching servers near what the content is most frequently accessed.  ***What is SQL Azure?*** A cloud-based relational database service with SQL Azure Reporting. 57](#_Toc521765711)

[***What is the use of Window Azure SDK?*** With Windows Azure SDK, a Visual Studio programmer can employ Cloud project template, develop Windows Azure applications, and securely publish an application to cloud using certificate. 58](#_Toc521765712)

[Design Pattern 58](#_Toc521765713)

[Dependency Injector 58](#_Toc521765714)

[Understanding Inversion of Control, Dependency Injection and Service Locator 58](#_Toc521765715)

[IoC and DI 58](#_Toc521765716)

[Dependency Injection (DI) 59](#_Toc521765717)

[Dependency Injection VS Service Locator 59](#_Toc521765718)

[Simple Dependency Injection 59](#_Toc521765719)

[Service Locator 61](#_Toc521765720)

[Repository pattern 61](#_Toc521765721)

[*What is Inversion of Control?* 62](#_Toc521765722)

[**Abstract Factory Pattern:** 64](#_Toc521765723)

[**Builder Pattern:** Separate representation and object construction 68](#_Toc521765724)

[**Factory Method Pattern:** 68](#_Toc521765725)

[Define an interface for creating an object, but let subclasses decide which class to instantiate. Factory Method lets a class defer instantiation to subclasses. 68](#_Toc521765726)

[**Prototype Pattern Design Pattern** 68](#_Toc521765727)

[Clone or copy initialized instances 68](#_Toc521765728)

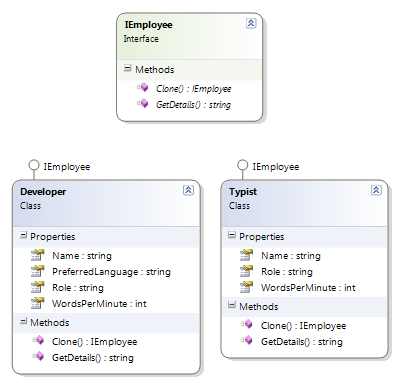
[**Singleton Pattern Design Pattern** 68](#_Toc521765729)

[**Difference between static class and singleton pattern?** 69](#_Toc521765730)

[Structural Patterns 69](#_Toc521765731)

[Adapter Pattern: 69](#_Toc521765732)

[What is Prototype Pattern? 69](#_Toc521765733)

[ 70](#_Toc521765734)

[ **Bridge Pattern::** Separate implementation and object interfaces 73](#_Toc521765735)

[ **Composite:** Simple and composite objects tree 73](#_Toc521765736)

[ **Decorator:** Dynamically add responsibilities to objects 73](#_Toc521765737)

[ **Facade:** Class that represents subclasses and subsystems 73](#_Toc521765738)

[ **Flyweight:** Minimize memory usage by sharing as much data as possible with similar objects 73](#_Toc521765739)

[ **Proxy:** Object that represents another object 73](#_Toc521765740)

[Behavioral Patterns 73](#_Toc521765741)

[Chain of Responsibility: 73](#_Toc521765742)

[ Pass requests between command and processing objects within a chain of objects 73](#_Toc521765743)

[Command: 73](#_Toc521765744)

[ Encapsulate a method call as an object containing all necessary information 73](#_Toc521765745)

[Interpreter: 73](#_Toc521765746)

[ Include language elements and evaluate sentences in a given language 73](#_Toc521765747)

[Iterator pattern : 73](#_Toc521765748)

[ 75](#_Toc521765749)

[Mediator Pattern: 75](#_Toc521765750)

[Memento Pattern : 75](#_Toc521765751)

[Observer Pattern : 75](#_Toc521765752)

[ Notify dependent objects of state changes 75](#_Toc521765753)

[State Pattern: 75](#_Toc521765754)

[ 76](#_Toc521765755)

[ **Strategy Design Pattern** 76](#_Toc521765756)

[ **Template Method:** Define an algorithm skeleton and delegate algorithm steps to subclasses so that they may be overridden 79](#_Toc521765757)

[ **Visitor:** Add new operations to classes without modifying them 79](#_Toc521765758)

[What is the difference between Factory and Abstract Factory Patterns? 79](#_Toc521765759)

[Project Management 79](#_Toc521765760)

[Project Estimation 79](#_Toc521765761)

[Stable Activities 80](#_Toc521765762)

[Reference 83](#_Toc521765763)

[http://www.mindtools.com/pages/article/newPPM\_01.htm 83](#_Toc521765764)

# HTML 5

## New Elements in HTML5

## HTML5 offers new elements for better document structure:

|  |  |
| --- | --- |
| Tag | Description |
| <article> | Defines an article in the document |
| <aside> | Defines content aside from the page content |
| <bdi> | Defines a part of text that might be formatted in a different direction from other text |
| <details> | Defines additional details that the user can view or hide |
| <dialog> | Defines a dialog box or window |
| <figcaption> | Defines a caption for a <figure> element |
| <figure> | Defines self-contained content, like illustrations, diagrams, photos, code listings, etc. |
| <footer> | Defines a footer for the document or a section |
| <header> | Defines a header for the document or a section |
| <main> | Defines the main content of a document |
| <mark> | Defines marked or highlighted text |
| <menuitem> | Defines a command/menu item that the user can invoke from a popup menu |
| <meter> | Defines a scalar measurement within a known range (a gauge) |
| <nav> | Defines navigation links in the document |
| <progress> | Defines the progress of a task |
| <rp> | Defines what to show in browsers that do not support ruby annotations |
| <rt> | Defines an explanation/pronunciation of characters (for East Asian typography) |
| <ruby> | Defines a ruby annotation (for East Asian typography) |
| <section> | Defines a section in the document |
| <summary> | Defines a visible heading for a <details> element |
| <time> | Defines a date/time |
| <wbr> | Defines a possible line-break |

## New Semantic Elements in HTML5

Many web sites contain HTML code like: <div id="nav"> <div class="header"> <div id="footer">  
to indicate navigation, header, and footer.

HTML5 offers new semantic elements to define different parts of a web page:

* <header>
* <nav>
* <section>
* <article>
* <aside>
* <figure>
* <figcaption>
* <footer>
* <details>
* <summary>
* <mark>
* <time>



# JavaScript

JavaScript (JS) is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. It has also become common in server-side programming, game development and the creation of desktop applications.

### jQuery:

jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery is free, open source software, licensed under the MIT License. jQuery’s syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme-able widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and web applications. Used by over 80% of the 10,000 most visited websites, jQuery is the most popular JavaScript library in use today

# Framework

## What is concept of Boxing and Unboxing ?

Boxing permits any value type to be implicitly converted to type object or to any interface type implemented by value type. Boxing is a process in which object instances are created and copy values in to that instance. Unboxing is vice versa of boxing operation where the value is copied from the instance in to appropriate storage location.

## What is the difference between System exceptions and Application exceptions?

All exception derives from Exception Base class. Exceptions can be generated programmatically or can be generated by system. Application Exception serves as the base class for all application specific exception classes. It derives from Exception but does not provide any extended functionality. You should derive your custom application exceptions from Application Exception. Application exception are used when we want to define user defined exception. While system exception are all which are defined by .NET

## Access Modifier

**Public**

The type or member can be accessed by any other code in the same assembly or another assembly that references it.

**private**

The type or member can only be accessed by code in the same class or struct.

**protected**

The type or member can only be accessed by code in the same class or struct, or in a derived class.

**internal**

The type or member can be accessed by any code in the same assembly, but not from another assembly.

**protected internal**

The type or member can be accessed by any code in the same assembly, or by any derived class in another assembly.

[**Anonymous Type vs Dynamic Type**](http://stackoverflow.com/questions/391529/anonymous-type-vs-dynamic-type)

An anonymous type is a real, **compiler-generated type that is created for you**. The good thing about this is that the compiler can re-use this type later for other operations that require it as it is a POCO.

My understanding of dynamic types is that **they are late-bound, meaning** that the CLR (or DLR) will evaluate the object at execution time and then use duck typing to allow or disallow member access to the object.

So I guess the difference is that anonymous types are true POCOs that the compiler can see but you can only use and dynamic types are late-bound dynamic objects.

# ASP.NET

**What is the difference between Server.Transfer and response.Redirect ?**

Following are the major differences between them : -

Response.Redirect sends message to the browser saying it to move to some different page, while server.transfer does not send any message to the browser but rather redirects the user directly from the server itself.

So in server.transfer there is no round trip while response.redirect has a round trip and hence puts a load on server.

Using Server.Transfer you can not redirect to a different from the server itself. Example if your server is www.yahoo.com you can use server.transfer to move to www.microsoft.com but yes you can move to www.yahoo.com/travels, i.e. within websites. This cross server redirect is possible only using Response.redirect.

With server. Transfer you can preserve your information. It has a parameter called as “preserveForm”. So the existing query string etc. will be able in the calling page. In response.redirect you can maintain the state, but has lot of drawbacks.

**Describe state management in ASP.NET.**

State management is a technique to manage a state of an object on different request.

The HTTP protocol is the fundamental protocol of the World Wide Web. HTTP is a stateless protocol means every request is from new user with respect to web server. HTTP protocol does not provide you with any method of determining whether any two requests are made by the same person.

Maintaining state is important in any web application. There are two types of state management system in ASP.NET.

**Client-side state management**

**Server-side state management**

**Explain client side state management system.**

ASP.NET provides several techniques for storing state information on the client. These include the following:

**View state :** ASP.NET uses view state to track values in controls between page requests. It works within the page only. You cannot use view state value in next page.

**Control state:** You can persist information about a control that is not part of the view state. If view state is disabled for a control or the page, the control state will still work.

**Hidden fields:** It stores data without displaying that control and data to the user’s browser. This data is presented back to the server and is available when the form is processed. Hidden fields data is available within the page only (page-scoped data).

**Cookies:** Cookies are small piece of information that server creates on the browser. Cookies store a value in the user’s browser that the browser sends with every page request to the web server.

**Query strings:** In query strings, values are stored at the end of the URL. These values are visible to the user through his or her browser’s address bar. Query strings are not secure. You should not send secret information through the query string.

**Explain server side state management system.**

The following objects are used to store the information on the server:

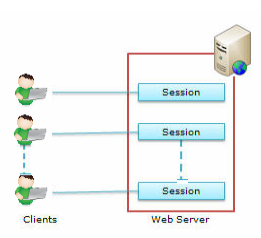
**Application State:** This object stores the data that is accessible to all pages in a given Web application. The Application object contains global variables for your ASP.NET application.

**Cache Object:** Caching is the process of storing data that is used frequently by the user. Caching increases your application’s performance, scalability, and availability. You can catch the data on the server or client.

**Session State:** Session object stores user-specific data between individual requests. This object is same as application object but it stores the data about particular user.

## What is Session object? Describe in detail

HTTP is a stateless protocol; it can't hold the user information on web page. If user inserts some information, and move to the next page, that data will be lost and user would not able to retrieve the information. For accessing that information we have to store information. Session provides that facility to store information on server memory. It can support any type of object to store. For every user Session data store separately means session is user specific.



Storing the data in Session object.

**Session [“message”] = “Hello World!”;**

Retrieving the data from Session object.

**Label1.Text = Session[“message”].ToString();**

What are the Advantages and Disadvantages of Session?

Following are the basic advantages and disadvantages of using session.

**Advantages:**

It stores user states and data to all over the application.

Easy mechanism to implement and we can store any kind of object.

Stores every user data separately.

Session is secure and transparent from user because session object is stored on the server.

**Disadvantages:**

Performance overhead in case of large number of user, because of session data stored in server memory.

Overhead involved in serializing and De-Serializing session Data. Because In case of State Server and SQLServer session mode we need to serialize the object before store.

## Differentiate between a HyperLink control and a LinkButton control.

A HyperLink control does not have the Click and Command events while the LinkButton control has them, which can be handled in the code-behind file of the Web page.

## What are the various types of validation controls provided by ASP.NET?

ASP.NET provides 6 types of validation controls as listed below:

RequiredFieldValidator - It is used when you do not want the container to be empty. It checks if the control has any value or not.   
 RangeValidator - It checks if the value in validated control is within the specified range or not.   
 CompareValidator - Checks if the value in controls matches some specific values or not.   
 RegularExpressionValidator - Checks if the value matches a specific regular expression or not.  
 CustomValidator - Used to define User Defined validation.  
 Validation Summary -Displays summary of all current validation errors on an ASP.NET page.

## What is caching?

Caching is the technique of storing frequently used items in memory so that they can be accessed more quickly.   
By caching the response, the request is served from the response already stored in memory.   
It’s important to choose the items to cache wisely as Caching incurs overhead.   
A Web form that is frequently used and does not contain data that frequently changes is good for caching.   
A cached web form freezes form’s server-side content and changes to that content do not appear until the cache is refreshed.

## Explain the use of duration attribute of @OutputCache page directive.

The @OutputCache directive’s Duration attribute determines how long the page is cached.   
If the duration attribute is set to 60 seconds, the Web form is cached for 60 seconds; the server loads the response in memory and retains that response for 60 seconds.   
Any requests during that time receive the cached response.   
Once the cache duration has expired, the next request generates a new response and cached for another 60 seconds.

**Describe the Events in the Life Cycle of a Web Application**

A web application starts when a browser requests a page of the application first time. The request is received by the IIS which then starts ASP.NET worker process (aspnet\_wp.exe). The worker process then allocates a process space to the assembly and loads it. An application\_start event occurs followed by Session\_start. The request is then processed by the ASP.NET engine and sends back response in the form of HTML. The user receives the response in the form of page.  
  
The page can be submitted to the server for further processing. The page submitting triggers postback event that causes the browser to send the page data, also called as view state to the server. When server receives view state, it creates new instance of the web form. The data is then restored from the view state to the control of the web form in Page\_Init event.  
  
The data in the control is then available in the Page\_load event of the web form. The cached event is then handled and finally the event that caused the postback is processed. The web form is then destroyed. When the user stops using the application, Session\_end event occurs and session ends. The default session time is 20 minutes. The application ends when no user accessing the application and this triggers Application\_End event. Finally all the resources of the application are reclaimed by the Garbage collector.

## Describe the application event handlers in ASP.NET

Following are the application event handlers:  
  
Application\_Start: This event occurs when the first user visits a page of the application.  
Application\_End: This event occurs when there are no more users of the application.  
Application\_BeginRequest: This occurs at the beginning of each request to the server.  
Application\_EndRequest: occurs at the end of each request to the server.  
Session\_Start: This event occurs every time when any new user visits.  
Session\_End: occurs when the users stop requesting pages and their session times out.

## What are the Web Form Events available in ASP.NET?

Page\_Init  
Page\_Load  
Page\_PreRender  
Page\_Unload  
Page\_Disposed  
Page\_Error  
Page\_AbortTransaction  
Page\_CommitTransaction  
Page\_DataBinding

## What are the options in ASP.NET to maintain state?

**Client-side state management** : This maintains information on the client’s machine using Cookies, View State, and Query Strings.  
**Cookies** :A cookie is a small text file on the client machine either in the client’s file system or memory of client browser session. Cookies are not good for sensitive data. Moreover, Cookies can be disabled on the browser. Thus, you can’t rely on cookies for state management.  
  
**View State** :Each page and each control on the page has View State property. This property allows automatic retention of page and controls state between each trip to server. This means control value is maintained between page postbacks. Viewstate is implemented using \_VIEWSTATE, a hidden form field which gets created automatically on each page. You can’t transmit data to other page using view state.  
**Querystring**:Query strings can maintain limited state information. Data can be passed from one page to another with the URL but you can send limited size of data with the URL. Most browsers allow a limit of 255 characters on URL length.  
**Server-side state management** :This kind of mechanism retains state in the server.  
**Application State** :The data stored in the application object can be shared by all the sessions of the application. Application object stores data in the key value pair.  
**Session State** : Session State stores session-specific information and the information is visible within the session only. ASP.NET creates unique sessionId for each session of the application. SessionIDs are maintained either by an HTTP cookie or a modified URL, as set in the application’s configuration settings. By default, SessionID values are stored in a cookie.  
**Database** : Database can be used to store large state information. Database support is used in combination with cookies or session state.

## What is delegate?

A delegate acts like a strongly type function pointer. Delegates can invoke the methods that they reference without making explicit calls to those methods. It is type safe since it holds reference of only those methods that match its signature. Unlike other classes, the delegate class has a signature. Delegates are used to implement event programming model in .NET application. Delegates enable the methods that listen for an event, to be abstract.

## How you can access the Properties and Controls of Master Pages from content pages?

You can access the Properties and Controls of Master Pages from content pages. In many situations you need User’s Name in different content pages. You can set this value inside the master page and then make it available to content pages as a property of the master page.

We will follow the following steps to reference the properties of master page from content pages.

**Step: 1**

Create a property in the master page code-behind file.

public String UserName   
{  
get {  
    return (String)Session["Name"];  
}  
set {  
Session ["Name"] = value;  
}  
}

**Step: 2**

Add the @ MasterTypedeclaration to the .aspx content page to reference master properties in a content page. This declaration is added just below the @ Page declaration as follows:

**<%@ Page Title=" TEST" Language="C#" MasterPageFile="~/CareerRide.master" AutoEventWireup="true" CodeFile="CareerRideWelcome.aspx.cs" Inherits="CareerRideWelcome" %>**

**<%@ MasterTypeVirtualPath="~/CareerRide.master" %>**

**Step: 3**

Once you add the @ MasterType declaration, you can reference properties in the master page using the Master class. For example take a label control that id is ID="Label1"

**Label1.Text= Master.UserName ;**

**For referencing controls in the Master Page we will write the following code.**

**Content Page Code.**

**protected void Button1\_Click(object sender, EventArgs e)**  
**{**  
**TextBox txtName= (TextBox)Master.FindControl("TextBox1");**  
**Label1.Text=txtName.Text;**   
**}**

To reference controls in a master page, call **Master.FindControl** from the content page.

## What is reflection?

Reflection is a mechanism through which types defined in the metadata of each module can be accessed. The System.Reflection namespaces contains classes that can be used to define the types for an assembly.

## What is AutoPostback?

AutoPostBack automatically posts the page back to the server when state of the control is changed.

## How to turn off cookies for a page?

Cookie.Discard Property when true, instructs the client application not to save the Cookie on the user's hard disk when a session ends.

## Response.Redirect vs Server.Transfer

Server.Transfer is only applicable for aspx files. It transfers page processing to another page without making round-trip back to the client's browser. Since no round trips, it offers faster response and doesn't update client url history list.  
  
Response.Redirect is used to redirect to another page or site. This performs a trip back to the client where the client’s browser is redirected to the new page.

## List down the sequence of methods called during the page load.

Init() - Initializes the page.  
Load() - Loads the page in the server memory.  
PreRender() - the brief moment before the page is displayed to the user as HTML  
Unload() - runs just after page finishes loading.

## Define authentication and authorization.

**Authorization**: The process of granting access privileges to resources or tasks within an application.  
**Authentication**: The process of validating the identity of a user

## ASP.NET Performance

Reference : <http://msdn.microsoft.com/en-us/library/vstudio/cc668225(v=vs.100).aspx>

**Page and Server Control Processing**

Avoid unnecessary round trips to the server   In some situations, you can use Microsoft Ajax and partial-page rendering to accomplish tasks in browser code without performing a full postback. For example, you can use Ajax features to validate user input in the browser before the input is submitted to the server. For more information, see [Microsoft Ajax Overview](http://msdn.microsoft.com/en-us/library/vstudio/bb398874(v=vs.100).aspx) and [Partial-Page Rendering Overview](http://msdn.microsoft.com/en-us/library/vstudio/bb386573(v=vs.100).aspx).

If you develop custom server controls, consider designing them to render client script for some of their functionality. This can significantly reduce the number of times that information is sent to the Web server. For more information, see [Developing Custom ASP.NET Server Controls](http://msdn.microsoft.com/en-us/library/vstudio/zt27tfhy(v=vs.100).aspx) and [Creating Custom Client Script by Using the Microsoft Ajax Library](http://msdn.microsoft.com/en-us/library/vstudio/bb386453(v=vs.100).aspx).

Use the Page object's IsPostBack property to avoid unnecessary processing   Avoid running code on each postback if it only has to run the first time the page is requested. You can test the [IsPostBack](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.page.ispostback(v=vs.100).aspx) property to conditionally execute code depending on whether the page is running in response to a server control event.

Leave buffering on unless you have a specific reason to turn it off   There is a significant performance cost for disabling buffering of ASP.NET Web pages. For more information, see the [Buffer](http://msdn.microsoft.com/en-us/library/vstudio/system.web.httpresponse.buffer(v=vs.100).aspx) property.

Use the [Transfer](http://msdn.microsoft.com/en-us/library/vstudio/system.web.httpserverutility.transfer(v=vs.100).aspx) method of the [Server](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.page.server(v=vs.100).aspx) object or use cross-page posting to redirect between ASP.NET pages in the same application   For more information, see [Redirecting Users to Another Page](http://msdn.microsoft.com/en-us/library/vstudio/x3x8t37x(v=vs.100).aspx).

**State Management**

**Save server control view state only when it is required**   View state enables server controls to repopulate property values on a round trip without requiring you to write code. However, view state affects performance and page size because it is passed to and from the server in a hidden form field

**<%@ Page EnableViewState="false" %>**

**<asp:datagrid EnableViewState="false" datasource="..."**

**runat="server"/>**

**Avoid using view state encryption unless you have to**   View state encryption prevents users from reading view-state values in the hidden view-state form field. For example, you might encrypt view state if a page includes a [GridView](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.gridview(v=vs.100).aspx) control that maintains an identifier field in the [DataKeyNames](http://msdn.microsoft.com/en-us/library/vstudio/system.web.ui.webcontrols.gridview.datakeynames(v=vs.100).aspx) property in order to coordinate updates to records. Because you do not want the identifier to be visible to users, you can encrypt view state. However, encryption has a constant performance cost for initialization, and it has an additional cost that depends on the size of view state that is being encrypted. Encryption is performed every time that the page is loaded. Therefore, the same performance effect occurs when the page is first requested and during every postback.

**Disable session state when you are not using it**   To disable session state for a page, set the [EnableSessionState](http://msdn.microsoft.com/en-us/library/vstudio/system.web.configuration.pagessection.enablesessionstate(v=vs.100).aspx) attribute in the [@ Page](http://msdn.microsoft.com/en-us/library/vstudio/ydy4x04a(v=vs.100).aspx) directive to false, as in the following example:

**Choose the appropriate session-state provider for your** application   ASP.NET provides multiple ways to store session data for your application. These include in-process session state, out-of-process session state as a Windows service, and out-of-process session state in a SQL Server database. (You can also create a custom session state provider to store session data in a data store that you specify.) Each approach has advantages, but in-process session state is by far the fastest approach

**Data Access**

**Use SQL Server and stored procedures for data access**   SQL Server is the recommended choice for data storage to create high-performance, scalable Web applications. When you use the managed SQL Server provider, you can get an additional performance boost by using compiled stored procedures wherever possible instead of using SQL commands

Cache data and page output whenever possible   Use ASP.NET caching if pages or data do have to be dynamically computed for every page request. If possible, design pages and data requests for caching, especially for situations where you expect heavy traffic. By using the cache appropriately, you can improve the performance of your site more than by using any other feature of the .NET Framework.

# Json

**Why Json?**

Ubiquity

Simplicity

Readability

# LINQ

## [LINQ Single vs SingleOrDefault vs First vs FirstOrDefault](http://www.technicaloverload.com/linq-single-vs-singleordefault-vs-first-vs-firstordefault/)

Many people get confused about the difference between [Single](http://msdn.microsoft.com/en-us/library/system.linq.enumerable.single.aspx), [SingleOrDefault](http://msdn.microsoft.com/en-us/library/system.linq.enumerable.singleordefault.aspx), [First](http://msdn.microsoft.com/en-us/library/system.linq.enumerable.first.aspx), and [FirstOrDefault](http://msdn.microsoft.com/en-us/library/system.linq.enumerable.firstordefault.aspx)methods in Linq. Below is a chart explaining the difference between them and examples of each scenario.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Single() | SingleOrDefault() | First() | FirstOrDefault() |
| Description | Returns a single, specific element of a sequence | Returns a single, specific element of a sequence, or a default value if that element is not found | Returns the first element of a sequence | Returns the first element of a sequence, or a default value if no element is found |
| Exception thrown when | There are 0 or more than 1 elements in the result | There is more than one element in the result | There are no elements in the result | Only if the source is null (they all do this) |
| When to use | If exactly 1 element is expected; not 0 or more than 1 | When 0 or 1 elements are expected | When more than 1 element is expected and you want only the first | When more than 1 element is expected and you want only the first. Also it is ok for the result to be empty |

# Entity Framework

## Advantage

ORM's exist to simplify and reduce the workload on the developer. In many well-designed relational models, the database can be accessed via code that is highly repetitive (just change a table name, column names, but the behavior is very similar). An ORM generates code that the developer can use right away, and tweak as necessary. The developer does not need to write SQL - that is the job of the ORM now. It also refreshes the generated code by automatically adjusting to database schema changes

## [Entity Framework 4 vs NHibernate](http://stackoverflow.com/questions/1639043/entity-framework-4-vs-nhibernate)

Update: I haven't used Entity Framework since version 4.0, so my answer can be outdated. I'm still using NH or pure ADO .NET in my projects. And I don't even want to look at what's new in EF since 4.0, because NH works perfectly.

Actually is pretty easy to compare them when you have used both. There are some serious limitations with EF4, I can name some which I encountered by myself:

EF4 problems:

Eager Loading and shaping the result: EF4 eager loading system (Include("Path")) generates improper SQL, with looping JOIN's , which will execute thousands(not literally) time slower for many-to-many relationships then hand written SQL (it's effectively unusable).

Materializer can't materialize associated entities: If you can think you can overcome previous problem by providing you own SQL query, you are wrong. EF4 can't materialize(map) associated entities from JOIN SQL query, it can only load data from one table (So if you have Order.Product, SELECT \* FROM order LEFT JOIN Product will initialize only Order object, Product will remain null, thought all necessary data is fetched in query to init it ). This can be overcome by using EFExtensions community add-on, but the code you will have to write for this is really ugly (I tried).

Self-Tracking Entities: Many say that Self-tracking entities are cool for N-tier development including the top answer in this thread. Thought I haven't even give them a try, I can say they are not. Every input can be forged, you can't simply take the changes that user sends you and apply them to data base, why not give the user direct data base access then? Any way you will have to load the data user is about to change from DB, check that it exist not exists do permissions checks etc etc. You can't trust user on the state of entity he is sending to server, you will anyway have to load this entity from DB and determine it's state and other things, so this information is useless, as do Self-Tracking entities unless you do a private trusted n-tier system for internal use, in which case maybe you could give just plain DB access. (Thats my thoughts about ST Entities and N-tire, I'm not very expericned in N-Tier, so it can change, if I misunderstood something here comment it)

Logging, Events, integrating business logic: EF4 is like black box, it do something and you have no idea what it do. There is only one event OnSavingChanges where you can put some business logic you need to run before something happens with DB, and if you need to apply some changes to business objects before something happens you will have to dig in ObjectStateManager, and this is really ugly, code can become huge. If you for example using Repository pattern and what to be notified on changes made to DB in clean object way, you will have hard time doing this with EF.

**Extensibility**: All EF code is private and internal, if you don't like something (and you will not like a LOT if you are serious about EF using), no way you will change this in easy way, In fact I'm sure it's easier to write you own ORM from scratch (I did) then make EF work as you need. As example take a look at Extensions source, it's based on extensions methods, and different "hacks" to make EF little more usable, and the code is pretty ugly (and it's not authors fault, when everything in EF is private this is the only way to extend it).

I can continue to write bad things about EF and how painful it was for me to work with it for like 20 pages, and maybe I will.

What about NHibernate? It's absolutely different level, it's like comparing PHP to C#, EF4 is like in Stone-age, it's like EF is 10 years behind then NHibernate in development progress, and in fact it is, Hibernate was started in 2001. If you have free time to learn and switch on Nhibernate, do it

## Using Code First model, how can I mark a field/property as the primary key if it does not follow the code first convention?

In our case above, EF looks for the word “ID” with a combination with the entity name (e.g. Project) to determine both the EntityKey and the primary key. If we rename the “Id” to say “UniqueProjectIdentifier”, we will need to decorate that property with the KeyAttribute ([Key]) to make it all work.

In the code below, we redefined our primary key but did not provide any data annotations.

public class Project  
{  
 // Code First has to be told that   
 // this as the primary key column  
 [Key]  
 public int UniqueProjectIdentifier { get; set; }  
 // this becomes a nullable column  
 public string Name { get; set; }  
 public string Description { get; set; }  
  
 // list of tasks for a project  
 public virtual List<Task> Tasks { get; set; }  
}

## When you have a annotate a property as Primary key in a table, how do you enable foreign key relationship from another table?

Now that we have a custom primary key, we also have to annotate a foreign key for the Task table. The solution is to define a navigation property for Task and annotate it to mark the ProjectId property as the FK.

// Code First infers this as the primary key column  
 public int TaskId { get; set; }  
 public string Name { get; set; }  
  
 public DateTime StartDate { get; set; }  
 public DateTime EndDate { get; set; }  
  
 // this is inferred as Foreign key to project table  
 public int ProjectId { get; set; }  
  
 // explicitly define the FK  
 [ForeignKey("ProjectId")]  
 public virtual Project Project { get; set; }

How do you mark a property as required? For example, For a Project, the Name is a required field.

You use the [Required] attribute to mark a property as required.

public class Project  
{  
 // Code First has to be told that   
 // this as the primary key column  
 [Key]  
 public int UniqueProjectIdentifier { get; set; }  
 // this becomes a non-nullable column  
 [Required]  
 public string Name { get; set; }  
 public string Description { get; set; }  
  
 // list of tasks for a project  
 public virtual List<Task> Tasks { get; set; }  
}

How do you enforce a field to have a minimum and maximum number of characters? For example, the Description on a Project should be a minimum of 10 and a maximum of 500?

EF provides us with convenient property annotations of MinLength and maxLength.

public class Project  
{  
 // Code First has to be told that   
 // this as the primary key column  
 [Key]  
 public int UniqueProjectIdentifier { get; set; }  
 // this becomes a non-nullable column  
 [Required]  
 public string Name { get; set; }  
 [MaxLength(500, ErrorMessage="Maximum of 500 characters please")]  
 [MinLength(10, ErrorMessage="Minimum of 10 characters required")]  
 public string Description { get; set; }  
  
 // list of tasks for a project  
 public virtual List<Task> Tasks { get; set; }  
}

Define a property in project class named ProjectCode that is not mapped to the database. ProjectCode is internally calculated as a combination of project ID and Title.

Normally, in code first convention, all properties are mapped to the database. If we want to exclude a specific property (generally a computed property), we can annotate it with [NotMapped] attribute.

public class Project  
{  
 // Code First has to be told that   
 // this as the primary key column  
 [Key]  
 public int UniqueProjectIdentifier { get; set; }  
 // this becomes a non-nullable column  
 [Required]  
 public string Name { get; set; }  
 [MaxLength(500, ErrorMessage="Maximum of 500 characters please")]  
 [MinLength(10, ErrorMessage="Minimum of 10 characters required")]  
 public string Description { get; set; }  
  
 // list of tasks for a project  
 public virtual List<Task> Tasks { get; set; }  
  
 [NotMapped]  
 public string ProjectCode  
 {  
 get  
 {  
 return UniqueProjectIdentifier + Name;  
 }  
 }  
}

Define a property in project class named ProjectCode that is not mapped to the database. ProjectCode is internally calculated as a combination of project ID and Title.

Normally, in code first convention, all properties are mapped to the database. If we want to exclude a specific property (generally a computed property), we can annotate it with [NotMapped] attribute.

public class Project  
{  
 // Code First has to be told that   
 // this as the primary key column  
 [Key]  
 public int UniqueProjectIdentifier { get; set; }  
 // this becomes a non-nullable column  
 [Required]  
 public string Name { get; set; }  
 [MaxLength(500, ErrorMessage="Maximum of 500 characters please")]  
 [MinLength(10, ErrorMessage="Minimum of 10 characters required")]  
 public string Description { get; set; }  
  
 // list of tasks for a project  
 public virtual List<Task> Tasks { get; set; }  
  
 [NotMapped]  
 public string ProjectCode  
 {  
 get  
 {  
 return UniqueProjectIdentifier + Name;  
 }  
 }  
}

How can you tell EF to have a different table or column name than that defined for the class?

By convention, EF defines the table and column names based on your class and property names. You can use the [Table] and [Column] annotations to tell EF to use different names.

[Table("ProjectItems")]  
public class Task  
{  
 // Code First infers this as the primary key column  
 public int TaskId { get; set; }  
 public string Name { get; set; }  
  
 [Column("CreationDate")]  
 public DateTime StartDate { get; set; }  
 public DateTime EndDate { get; set; }  
  
 // this is inferred as Foreign key to project table  
 public int ProjectId { get; set; }  
  
 // explicitly define the FK  
 [ForeignKey("ProjectId")]  
 public virtual Project Project { get; set; }  
}

For a date time property, how can you tell EF to automatically compute and insert the current date time when the row is created?

In our Project tasks, we want to automatically set the creation date when a new row is inserted. We can achieve this by telling EF that this property is a **[DatabaseGenerated**] property and that it is computed.

[Table("ProjectItems")]  
public class Task  
{  
 // Code First infers this as the primary key column  
 public int TaskId { get; set; }  
 public string Name { get; set; }  
  
 [Column("CreationDate")]  
 [DatabaseGenerated(DatabaseGeneratedOption.Computed)]  
 public DateTime StartDate { get; set; }  
 public DateTime EndDate { get; set; }  
  
 // this is inferred as Foreign key to project table  
 public int ProjectId { get; set; }  
  
 // explicitly define the FK  
 [ForeignKey("ProjectId")]  
 public virtual Project Project { get; set; }  
}

This code will throw an exception. Why? Because code first won't be able to determine the formula for the computed column. To solve this, you should only use this when pointing to existing databases OR use the [TimeStamp] column.

Another use of this attribute is when you do NOT want your primary key to be an auto incremented.

## When two tables have multiple relationships (for example, a task is created by employee 1 and updated by employee 2), who do you indicate which relationships go with which property?

Entity Framework provides us with [InverseProperty] attribute to indicate multiple relationships between two tables. Consider the following code first model where the Task class now has 2 pointers to Employee for CreatedBy and UpdatedBy. Also we have added an Employee class which has a list of tasks created and updated. NOTE that we have not (yet) added any data annotations to signify any inverse relationships. The goal is to show you how EF will not be able to recognize this.

public class Employee  
{  
 public int Id { get; set; }  
 public string name { get; set; }  
  
 [InverseProperty("CreatedBy")]  
 public List<Task> TasksCreated { get; set; }  
 [InverseProperty("UpdatedBy")]  
 public List<Task> TasksUpdated { get; set; }  
}

## Lezy load

One of the important functionality of Entity Framework is lazy loading. Lazy loading means delaying the loading of related data until you specifically request it. For example, Student class contains StudentAddress as complex property. So context first loads all the students from the database then it will load address of particular student when we access StudentAddress property as below.

using (var ctx = new SchoolDBEntities())

{

ctx.Configuration.LazyLoadingEnabled = true;

//Loading students only

IList<Student> studList = ctx.Students.ToList<Student>();

Student std = studList[0];

//Loads Student address for particular Student only (seperate SQL query)

StudentAddress add = std.StudentAddress;

}

## Eagerly loading related entities

Eager loading is the process whereby a query for one type of entity also loads related entities as part of the query. Eager loading is achieved by use of the Include method. For example, the queries below will load princesses and all the unicorns related to each princess.

using (var context = new UnicornsContext())

{

// Load all princesses and related unicorns

var princesses1 = context.Princesses

.Include(p => p.Unicorns)

.ToList();

// Load one princess and her related unicorns

var princess1 = context.Princesses

.Where(p => p.Name == "Cinderella")

.Include(p => p.Unicorns)

.FirstOrDefault();

// Load all princesses and related unicorns using a string

// to specify the relationship

var princesses2 = context.Princesses

.Include("Unicorns")

.ToList();

// Load one princess and her related unicorns using a string

// to specify the relationship

var princess2 = context.Princesses

.Where(p => p.Name == "Cinderella")

.Include("Unicorns")

.FirstOrDefault();

}

# OOPS

## Abstraction

It allows complex real world to be represented in simplified manner. Example color is abstracted to RGB. By just making the combination of these three colors we can achieve any color in world.It’s a model of real world or concept. Encapsulation It is a process of hiding all the internal details of an object from the outside world.

## Association

This is the simplest relationship between objects. Example every customer has sales. So Customer object and sales object have an association relation between them.

## Aggregation

This is also called as composition model. Example in order to make a “Accounts” class it has use other objects example “Voucher”, “Journal” and “Cash” objects. So accounts class is aggregation of these three objects.

## Inheritance

Hierarchy is used to define more specialized classes based on a preexisting generalized class. Example we have VEHICLE class and we can inherit this class make more specialized class like CAR, which will add new attributes and use some existing qualities of the parent class. Its shows more of a parent-child relationship. This kind of hierarchy is called inheritance.

## Polymorphism

When inheritance is used to extend a generalized class to a more specialized class, it includes behavior of the top class(Generalized class). The inheriting class often implements a behavior that can be somewhat different than the generalized class, but the name of the behavior can be same. It is important that a given instance of an object use the correct behavior, and the property of polymorphism allows this to happen automatically.

When you have a function defined in a class that you want to be implemented in an inherited class(es), you use virtual functions. The virtual functions could be implemented differently in different inherited class and the call to these functions will be decided at runtime.

using System;

namespace PolymorphismApplication {

class Shape {

protected int width, height;

public Shape( int a = 0, int b = 0) {

width = a;

height = b;

}

public virtual int area() {

Console.WriteLine("Parent class area :");

return 0;

}

}

class Rectangle: Shape {

public Rectangle( int a = 0, int b = 0): base(a, b) {

}

public override int area () {

Console.WriteLine("Rectangle class area :");

return (width \* height);

}

}

class Triangle: Shape {

public Triangle(int a = 0, int b = 0): base(a, b) {

}

public override int area() {

Console.WriteLine("Triangle class area :");

return (width \* height / 2);

}

}

class Caller {

public void CallArea(Shape sh) {

int a;

a = sh.area();

Console.WriteLine("Area: {0}", a);

}

}

class Tester {

static void Main(string[] args) {

Caller c = new Caller();

Rectangle r = new Rectangle(10, 7);

Triangle t = new Triangle(10, 5);

c.CallArea(r);

c.CallArea(t);

Console.ReadKey();

}

}

}

## What is difference between abstract classes and interfaces?

Following are the differences between abstract and interfaces :-

Abstract classes can have concrete methods while interfaces have no methods implemented.

Interfaces do not come in inheriting chain, while abstract classes come in inheritance.

## Generic Classes

Generic classes encapsulate operations that are not specific to a particular data type. The most common use for generic classes is with collections like linked lists, hash tables, stacks, queues, trees, and so on. Operations such as adding and removing items from the collection are performed in basically the same way regardless of the type of data being stored

**Array VS ArrayList**

Array (A System namespace) is a datatype, that can be used by calling indexes. During runtime, one cannot really change the size of the array, unless you use the method of copying the array and getting rid of the old one.

In .NET, the Visual Studio makes use of a special class to store the data. Because of this, the performance is actually quite fast. This is also because in an array, you need to specify the size and thus, the data is stored one after the other.

Examples:

**int[] myNumbers= new int[5];**

**myNumbers[0] = 16;**

ArrayList (System.Collections namespace) is a datatype collection. In order to fill an ArrayList, one can use the .Add property. ArrayLists are very dynamic in the sense that when you add and/or remove items from it, the performace stays the same.

The internal structure of an ArrayList is an array.

Examples:

ArrayList myArray = new ArrayList();

myArray.Add(“Steph”);

string str = myArray[0];

Most of the time, we tend to choose array lists rather than arrays since we have no idea how big it is going to turn out. Arrays are ideal when you know how many items you are going to put in it. Whenever possible, it is recommended to use arrays as this drastically improves the performance.

Array are sequence of homogeneous data while ArrayList is sequence of heterogenous data. That's why we have to typecast every data in ArrayLists.

Arrays are multidimensional but ArrayList is always single-dimensional.

Arrays are strongly typed, and work well as parameters. If you know the length of your collection and it is fixed, you should use an array.

ArrayLists are not strongly typed, every Insertion or Retrial will need a cast to get back to your original type. If you need a method to take a list of a specific type, ArrayLists fall short because you could pass in an ArrayList containing any type. ArrayLists use a dynamically expanding array internally, so there is also a hit to expand the size of the internal array when it hits its capacity

## Generic Data Type

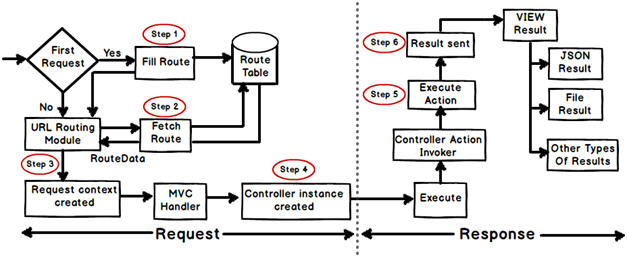
Generic data type create generic datatype instead of strongly datatype.

Decouple datatype in runtime.

# MVC

## Explain MVC application life cycle?

There are six broader events which occur in MVC application life cycle below diagrams summarize it.



Any web application has two main execution steps first understanding the request and depending on the type of the request sending out appropriate response. MVC application life cycle is not different it has two main phases first creating the request object and second sending our response to the browser.

**Creating the request object: -**The request object creation has four major steps. Below is the detail explanation of the same.

**Step 1 Fill route: -** MVC requests are mapped to route tables which in turn specify which controller and action to be invoked. So if the request is the first request the first thing is to fill the route table with routes collection. This filling of route table happens in the **global.asax** file.

**Step 2 Fetch route: -** Depending on the URL sent “UrlRoutingModule” searches the route table to create “RouteData” object which has the details of which controller and action to invoke.

**Step 3 Request context created: -** The “RouteData” object is used to create the “RequestContext” object.

**Step 4 Controller instance created: -** This request object is sent to “MvcHandler” instance to create the controller class instance. Once the controller class object is created it calls the “Execute” method of the controller class.

**Creating Response object: -** This phase has two steps executing the action and finally sending the response as a result to the view.

## HTTPGet

* **GET** - Requests data from a specified resource
* A hyperlink or anchor tag that points to an action will ALWAYS be an HttpGet.
* Data is submitted as a part of url.
* Data is visible to the user as it posts as query string.
* It is not secure but fast and quick.
* It use Stack method for passing form variable.
* Data is limited to max length of query string.
* It is good when you want user to bookmark page.

## HTTPPost

* POST - Submits data to be processed to a specified resource
* A Submit button will always initiate an HttpPost request.
* Data is submitted in http request body.
* Data is not visible in the url.
* It is more secured but slower as compared to GET.
* It use heap method for passing form variable
* It can post unlimited form variables.
* It is advisable for sending critical data which should not visible to users.

## What is ASP.NET MVC?

ASP.NET MVC is a web development framework from Microsoft that is based on MVC (Model-View-Controller) architectural design pattern. Microsoft has streamlined the development of MVC based applications using ASP.NET MVC framework

## [Difference between ASP.NET WebForms and ASP.NET MVC](http://www.devcurry.com/2009/09/difference-between-aspnet-webforms-and.html)

Here are some points that I wrote in an article for [www.dotnetcurry.com](http://www.dotnetcurry.com/) that differentiate ASP.NET WebForms from ASP.NET MVC:

|  |  |
| --- | --- |
| **ASP.NET WebForms** | **ASP.NET MVC** |
| Uses the ‘Page Controller’ pattern. Each page has a code-behind class that acts as a controller and is responsible for rendering the layout. | Uses the ‘Front Controller’ pattern. There is a single central controller for all pages to process web application requests and facilitates a rich routing architecture |
| Uses an architecture that combines the Controller (code behind) and the View (.aspx). Thus the Controller has a dependency on the View. Due to this, testing and maintainability becomes an issue. | ASP.NET MVC enforces a "separation of concerns". The Model does not know anything about the View. The View does not know there’s a Controller. This makes MVC applications easier to test and maintain |
| The View is called before the Controller. | Controller renders View based on actions as a result of the User Interactions on the UI. |
| At its core, you ‘cannot’ test your controller without instantiating a View. There are ways to get around it using tools. | At its core, ASP.NET MVC was designed to make test-driven development easier. You ‘can’ test your Controller without instantiating a View and carry out unit-tests without having to run the controllers in an ASP.NET process. |
| WebForms manage state by using view state and server-based controls. | ASP.NET MVC does not maintain state information by using view state |
| WebForms supports an event-driven programming style that is like Windows applications and is abstracted from the user. The State management is made transparent by using sessions, viewstate etc. In the process, the HTML output is not clean making it difficult to manage later. The ViewState also increases your page size. | In ASP.NET MVC, the output is clean and you have full control over the rendered HTML. The orientation is towards building standard compliant pages and provides full control over the behavior of an application. |
| Deep understanding of HTML, CSS and JavaScript is not required to a large extent since the WebForm model abstracts a lot of these details and provides automatic plumbing. While abstracting details to provide ease of use, sometimes a solution is overcomplicated, than it needs to be. | A thorough understanding of how HTML, CSS and JavaScript work together is required. The advantage is you can do a lot of jQuery and AJAX stuff in an efficient and simple manner than you would do in an ASP.NET application. |
| WebForms can drastically reduce time while building up intranet and internet applications that use a lot of controls (drag and drop model). Although this is true for development, a lot of time is spent later to code around limitations. | You lose the 'drag and drop' quick model of building your web applications. The focus is on control over the application behavior and test-driven development. The model is extensible and you do not have to spend time working around limitations. |
| Relatively simple to learn and pickup. Works very well for developers who initially have trouble with the HTTP/HTML model and are coming from a similar WinForms oriented event model. | There is a learning curve to understand the why, when and how of ASP.NET MVC. |
| Lesser amount of code is required to build webapps since a lot of components are integrated and provided out of the box. You can also use a lot of data controls provided out of the box that rely on ViewState. | Since the application tasks are separated into different components, amount of code required is more. Since ASP.NET MVC does not use ViewState, you cannot use Data controls like GridView, Repeater. |
| Works very well for small teams where focus is on rapid application development | Works well for large projects where focus in on testability and maintainability |

#### Can you please explain the request flow in ASP.NET MVC framework?

Request hits the controller coming from client. Controller plays its role and decides which model to use in order to serve the request further passing that model to view which then transforms the model and generates an appropriate response that is rendered to the client.

#### What is Routing in ASP.NET MVC?

In case of a typical ASP.NET application, incoming requests are mapped to physical files such as *.aspx* file. ASP.NET MVC framework uses friendly URLs that more easily describe user’s action but are not mapped to physical files.

ASP.NET MVC framework uses a routing engine that maps URLs to controller classes. We can define routing rules for the engine, so that it can map incoming request URLs to appropriate controller.

Practically, when a user types a URL in a browser window for an ASP.NET MVC application and presses “go” button, routing engine uses routing rules that are defined in *Global.asax* file in order to parse the URL and find out the path of corresponding controller.

#### What is the difference between ViewData, ViewBag and TempData?

In order to pass data from controller to view and in next subsequent request, ASP.NET MVC framework provides different options i.e., **ViewData, ViewBag and TempData**.

Both **ViewBag** and **ViewData** are used to communicate between controller and corresponding view. But this communication is only for server call, it becomes null if redirect occurs. So, in short, it's a mechanism to maintain state between controller and corresponding view.

**ViewData** is a dictionary object while ViewBag is a dynamic property (a new C# 4.0 feature). ViewData being a dictionary object is accessible using strings as keys and also requires typecasting for complex types. On the other hand, ViewBag doesn't have typecasting and null checks.

**TempData** is also a dictionary object that stays for the time of an HTTP Request. So, Tempdata can be used to maintain data between redirects, i.e., from one controller to the other controller.

#### What are Action Methods in ASP.NET MVC?

I already explained about request flow in ASP.NET MVC framework that request coming from client hits controller first. Actually MVC application determines the corresponding controller by using routing rules defined in *Global.asax*. And controllers have specific methods for each user actions. Each request coming to controller is for a specific Action Method. The following code example, “ShowBooks” is an example of an Action method.

public ViewResult ShowBooks(int id)

{

var computerBook = db.Books.Where(p => P.BookID == id).First();

return View(computerBook);

}

#### Explain the role of Model in ASP.NET MVC?

One of the core features of ASP.NET MVC is that it separates the input and UI logic from business logic. Role of Model in ASP.NET MVC is to contain all application logic including validation, business and data access logic except view, i.e., input and controller, i.e., UI logic.

Model is normally responsible for accessing data from some persistent medium like database and manipulate it.

#### What are Action Filters in ASP.NET MVC?

If we need to apply some specific logic before or after action methods, we use action filters. We can apply these action filters to a controller or a specific controller action. Action filters are basically custom classes that provide a means for adding pre-action or post-action behavior to controller actions.

For example:

* Authorize filter can be used to restrict access to a specific user or a role.
* OutputCache filter can cache the output of a controller action for a specific duration.

#### What are different ways of returning a View?

There are different ways for returning/rendering a view in MVC Razor.E.g. return View(), return RedirectToAction(), return Redirect() and return RedirectToRoute().

#### What is Html.Partial?

The Partial helper renders a partial view into a string. Typically, a partial view contains reusable markup you want to render from inside multiple different views. Partial has four overloads:

public void Partial(string partialViewName);

public void Partial(string partialViewName, object model);

public void Partial(string partialViewName, ViewDataDictionary viewData);

public void Partial(string partialViewName, object model,ViewDataDictionary viewData);

#### What is Html.RenderPartial?

The RenderPartial helper is similar to Partial, but RenderPartial writes directly to the response output stream instead of returning a string. For this reason, you must place RenderPartial inside a code block instead of a code expression. To illustrate, the following two lines of code render the same output to the output stream:

@{Html.RenderPartial("AlbumDisplay "); }

@Html.Partial("AlbumDisplay ")

#### If they are same then which one to use?

In general, you should prefer Partial to RenderPartial because Partial is more convenient (you don’t have to wrap the call in a code block with curly braces). However, RenderPartial may result in better performance because it writes directly to the response stream, although it would require a lot of use (either high site traffic or repeated calls in a loop) before the difference would be noticeable.

#### What are HTML Helpers?

HTML helpers are methods we can invoke on the Html property of a view. We also have access to URL helpers (via the Url property), and AJAX helpers (via the Ajax property). All these helpers have the same goal: to make views easy to author. The URL helper is also available from within the controller.Most of the helpers, particularly the HTML helpers, output HTML markup. For example, the BeginForm helper is a helper we can use to build a robust form tag for our search form, but without using lines and lines of code:

@using (Html.BeginForm("Search", "Home", FormMethod.Get)) {

#### <input type="text" name="q" />

<input type="submit" value="Search" />

}

#### What is ViewStart?

For group of views that all use the same layout, this can get a bit redundant and harder to maintain.

The \_ViewStart.cshtml page can be used to remove this redundancy. The code within this file

is executed before the code in any view placed in the same directory. This fi le is also recursively applied to any view within a subdirectory.

When we create a default ASP.NET MVC project, we find there is already a \_ViewStart.cshtml file in the Views directory. It specifies a default layout:

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

Because this code runs before any view, a view can override the Layout property and choose a different one. If a set of views shares common settings, the \_ViewStart.cshtml file is a useful place to consolidate these common view settings. If any view needs to override any of the common settings, the view can set those values to another value.

#### Explain Routing in MVC?

A route is a URL pattern that is mapped to a handler. The handler can be a physical

file, such as an .aspx file in a Web Forms application. Routing module is responsible for mapping incoming browser requests to particular MVC controller actions.

Routing within the ASP.NET MVC framework serves two main purposes:

* It matches incoming requests that would not otherwise match a file on the file system and maps the requests to a controller action.
* It constructs outgoing URLs that correspond to controller actions.

#### How route table is created in ASP.NET MVC?

When an MVC application first starts, the Application\_Start() method in global.asax is called. This method, calls the RegisterRoutes() method. The RegisterRoutes() method creates the route table for MVC application.

# Rest Web Service

## What is a RESTful Web Service?

A RESTful Web Service is a way of doing communication between client and server. Conceptually, using RESTful Web Services is very similar to web browsing. But instead of web pages we are working with data. Same HTTP protocol concepts are used. Writing RESTful Web Services are simpler than SOAP and are pretty much universally supported by all operating systems and programming languages. This allows for easy cross-platform integration.

• To create a resource on the server, use POST

• To retrieve a resource, use GET

• To change the state of a resource or to update it, use PUT

• To remove or delete a resource, use DELETE

## Enlist some of the HTTP methods with description.

Mentioned below is the list of HTTP methods with their descriptions:

**GET:** This is a read only operation which fetches the list of users on the server.

**PUT:** This operation is used for the creation of any new resource on the server.

**POST:** This operation is used for updating an old resource or for creating a new resource.

**DELETE:** As the name suggests, this operation is used for deleting any resource on the server.

**OPTIONS:** This operation fetches the list of any supported options of resources that are available on the server.

# WCF

### Binding

Binding will describes how client will communicate with service. There are different protocols available for the WCF to communicate to the Client. You can mention the protocol type based on your requirements.

A binding has several characteristics, including the following:

Transport -Defines the base protocol to be used like HTTP, Named Pipes, TCP, and MSMQ are some type of protocols.

Encoding (Optional) - Three types of encoding are available-Text, Binary, or Message Transmission Optimization Mechanism (MTOM). MTOM is an interoperable message format that allows the effective transmission of attachments or large messages (greater than 64K).

Protocol(Optional) - Defines information to be used in the binding such as Security, transaction or reliable messaging capability

The following table gives some list of protocols supported by WCF binding.

|  |  |
| --- | --- |
| Binding | Description |
| BasicHttpBinding | Basic Web service communication. No security by default |
| WSHttpBinding | Web services with WS-\* support. Supports transactions |
| WSDualHttpBinding | Web services with duplex contract and transaction support |
| WSFederationHttpBinding | Web services with federated security. Supports transactions |
| MsmqIntegrationBinding | Communication directly with MSMQ applications. Supports transactions |
| NetMsmqBinding | Communication between WCF applications by using queuing. Supports transactions |
| NetNamedPipeBinding | Communication between WCF applications on same computer. Supports duplex contracts and transactions |
| NetPeerTcpBinding | Communication between computers across peer-to-peer services. Supports duplex contracts |
| NetTcpBinding | Communication between WCF applications across computers. Supports duplex contracts and transactions |

### Contract

Collection of operation that specifies what the endpoint will communicate with outside world. Usually name of the Interface will be mentioned in the Contract, so the client application will be aware of the operations which are exposed to the client. Each operation is a simple exchange pattern such as one-way, duplex and request/reply.

#### Example:

Endpoints will be mentioned in the web.config file on the created service.

<system.serviceModel>

<services>

<service name="MathService"

behaviorConfiguration="MathServiceBehavior">

<endpoint

address="http://localhost:8090/MyService/MathService.svc" contract="IMathService"

binding="wsHttpBinding"/>

</service>

</services>

<behaviors>

<serviceBehaviors>

<behavior name="MathServiceBehavior">

<serviceMetadata httpGetEnabled="True"/>

<serviceDebug includeExceptionDetailInFaults="true" />

</behavior>

</serviceBehaviors>

</behaviors>

</system.serviceModel>

# Service Contract

Service contract describes the operation that service provide. A Service can have more than one service contract but it should have at least one Service contract.

Service Contract can be define using [**ServiceContract]** and [OperationContract] attribute. [ServiceContract] attribute is similar to the [WebServcie] attribute in the WebService and [OpeartionContract] is similar to the [WebMethod] in WebService

# Data Contract

**A data contract is a formal agreement between a service and a client that abstractly describes the data to be exchanged.**

Data contract can be explicit or implicit. Simple type such as int, string etc has an implicit data contract. User defined object are explicit or Complex type, for which you have to define a Data contract using [DataContract] and [DataMember] attribute.

It describes the external format of data passed to and from service operations

It defines the structure and types of data exchanged in service messages

It maps a CLR type to an XML Schema

t defines how data types are serialized and deserialized. Through serialization, you convert an object into a sequence of bytes that can be transmitted over a network. Through deserialization, you reassemble an object from a sequence of bytes that you receive from a calling application.

It is a versioning system that allows you to manage changes to structured data

# What is DataContractSerializer and How its different from XmlSerializer?

Serialization is the process of converting **an object instance to a portable and transferable format**. So, whenever we are talking about web services, serialization is very important.

Windows Communication Foundation has **DataContractSerializer** that is new in .NET 3.0 and uses opt-in approach as compared to XmlSerializer that uses opt-out. Opt-in means specify whatever we want to serialize while Opt-out means you don’t have to specify each and every property to serialize, specify only those you don’t want to serialize. **DataContractSerializer** is about 10% faster than XmlSerializer but it has almost no control over how the object will be serialized. If we wanted to have more control over how object should be serialized that XmlSerializer is a better choice.

### Please explain briefly different Instance Modes in WCF?

WCF will bind an incoming message request to a particular service instance, so the available modes are:

**Per Call**: instance created for each call, most efficient in term of memory but need to maintain session.

**Per Session**: Instance created for a complete session of a user. Session is maintained.

**Single**: Only one instance created for all clients/users and shared among all. Least efficient in terms of memory.

### What Message Exchange Patterns (MEPs) supported by WCF? Explain each of them briefly.

#### Request/Response

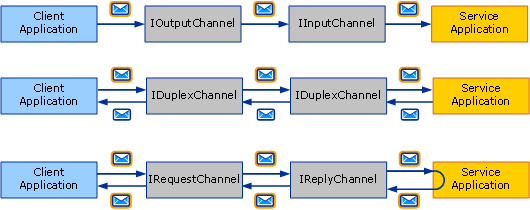
It’s the default pattern. In this pattern, a response message will always be generated to consumer when the operation is called, even with the void return type. In this scenario, response will have empty SOAP body.

#### One Way

In some cases, we are interested to send a message to service in order to execute certain business functionality but not interested in receiving anything back. OneWay MEP will work in such scenarios. If we want queued message delivery, OneWay is the only available option.

#### Duplex

The Duplex MEP is basically a two-way message channel. In some cases, we want to send a message to service to initiate some longer-running processing and require a notification back from service in order to confirm that the requested process has been completed.



# What are contracts in WCF? In WCF, all services expose contracts. The contract is a platform-neutral and standard way of describing what the service does. WCF defines four types of contracts.

**Service contracts**: Describe which operations the client can perform on the service.  
  
**Data contracts**: Define which data types are passed to and from the service. WCF defines implicit contracts for built-in types such as int and string, but we can easily define explicit opt-in data contracts for custom types.  
  
 **Fault contracts**: Define which errors are raised by the service, and how the service handles and propagates errors to its clients.  
  
**Message contracts**: Allow the service to interact directly with messages. Message contracts can be typed or untyped, and are useful in interoperability cases and when there is an existing message format we have to comply with.

Service Contract vs Data contract

The ServiceContract defines the service's contract - it's shape and form. It defines the name of the service, it's XML namespace etc., and it's typically an interface (but could also be applied to a class) that contains methods decorated with the [OperationContract] attribute - the service methods.

[ServiceContract]

public interface IMyService

{

[OperationContract]

Response GetData(int someKey);

}

The DataContract is a totally different beast - it decorates a class to define it as a class that's being used as a parameter or return value from one of the service methods. It's labelling that class as a "thing" to serialize onto the wire to transmit it. It's an instruction for the WCF runtime (the data contract serializer) that this class is intended to be used in a WCF service.

[DataContract]

public class Response

{

[DataMember]

int Key { get; set; }

[DataMember]

string ProductName { get; set; }

[DataMember]

DateTime DateOfPurchase { get; set; }

}

So the service contract and the data contract are two totally separate parts that play together to make a WCF service work - it's not like one could replace the other or something.

**Difference between using ChannelFactory and Proxies in WCF?**

A ChannelFactory creates a kind of Channel used by clients to communicate with service endpoints.

If we have control over Server and Client, then ChannelFactory is a good option because it relies on having local interfaces that actually describes the service i.e. service contract.  
On the other hand, If we don’t have control over server and only have WSDL/URL, then it’s better to generate proxy using Visual Studio or SvcUtil.  
SvcUtil is better option as compared to Visual Studio because we have more control in case of SvcUtil.

**What are the different ways to handle concurrency in WCF?**

There are three different ways to handle concurrency in WCF that are:  
a)    Single  
b)    Multiple  
c)    Reentrant  
Single: means at a given time, only a single request can be processed by WCF service instance. Other requests will be waiting until the first one is fully served.  
Multiple: means multiple requests can be served by multiple threads of a single WCF service instance.  
Reentrant: means a single WCF service instance can process one request at a given time but the thread can exit the service to call another service.  
We can apply these concurrency settings by putting ConcurrencyMode property in ServiceBehavior as follows:

**[ServiceBehavior(ConcurrencyMode = ConcurrencyMode.Multiple]**  
public class MyService : IMyService  
{  
}

### What is WCF throttling?

WCF throttling enables us to regulate the maximum number of WCF instances, concurrent calls and concurrent sessions. Basic purpose is to control our WCF service performance by using Service throttling behavior.

In configuration file we can set this behavior as follows:

*<serviceBehavior>*  
*<behavior name=”MyServiceBehavior”>*  
*<serviceThrottling*  
*maxConcurrentInstances=”2147483647”*  
*maxConcurrentCalls=”16″*  
*maxConcurrentSessions=”10″*  
*</behavior>*  
*</serviceBehavior>*

Above given values are the default ones, but we can update it after looking into the requirements of our application.

### Please explain about authorization options supported in WCF?

Authorization as a core feature of security in WCF supports different authorization types.

**Role-based authorization** is the most common authorization approach being used. In this approach, authenticated user has assigned roles and system checks and verifies that either a specific assigned role can perform the operation requested.

**Identity-based authorization** approach basically provides support for identity model feature which is considered to be an extension to role-based authorization option. In this approach, service verifies client claims against authorization policies and accordingly grant or deny access to operation or resource.

**Resource-based authorization** approach is a bit different because it’s applied on individual resources and secure those using windows access control lists (ACLs).

### What is Reliable Messaging in WCF?

We know that networks are not perfect enough and those might drop signals or in some scenarios there can be a possibility of wrong order of messages during message exchange.  
WCF allows us to ensure the reliability of messaging by implementing WS-ReliableMessaging protocol.  Here is how you can configure reliable messaging in WCF.

*<wsHttpBinding>*  
*<binding name=”Binding1″>*  
*<reliableSession*  
*enabled=”true”*  
*ordered=”true”*  
*inactivityTimeout=”00:02:00″ />*  
*</binding>*  
*</wsHttpBinding>*

### What are Reliable Sessions in WCF?

Reliable sessions actually ensure that the caller for messages will know about the lost message(s) but it can’t guarantee about the delivery of message(s).  
There is a misconception about reliable sessions that it ensures the session will never expire or stays for a very long time. This we can achieve by using timeout for sessions

# WCF Security

No transfer security mode

This ensure that no security is applied while communication between server and client

<wsHttpBinding >

<binding name="WCFSecurityExample">

<security mode="None"/>

</binding>

</wsHttpBinding>

## Transport security mode

When system is configured with ‘Transport’ mode, WCF uses secured communication protocol. The available secure transports are HTTPS, TCP, IPC and MSMQ. **Transport security encrypts all communication on the channel and provides integrity, privacy and mutual authentication**. It provides point-to-point security.

One of main disadvantage is that it can **only guarantee transfer security point-to-point, meaning it secure only at channel level. Message inside the channel will not get secured. In case of distributed communication, multiple intermediaries between service and client will not be secure.**

It is mainly used in intranet application

<wsHttpBinding >

<binding name="WCFSecurityExample">

<security mode="Transport"/>

</binding>

</wsHttpBinding>

<netTcpBinding>

<binding name="netTcpTransportBinding">

<security mode="Transport">

<Transport clientCredentialType="Windows" />

</security>

</binding>

</netTcpBinding>

## Message security mode

In this mode of configuration, message will get encrypted. Encrypting the message rather than transport enables the service to communicate securely over non secure transport such as HTTP. It provides end-to-end security.

One of the disadvantages of message security is that it may introduce call latency due to its inherent overhead.

It is mainly used in internet application.

<wsHttpBinding>

<binding name="wsHttpMessageBinding">

<security mode="Message">

<Message clientCredentialType="UserName" />

</security>

</binding>

</wsHttpBinding>

## Mixed transfer security mode

It uses Transport security for message integrity, privacy and service authentication and it uses Message security for securing client credential.

One of disadvantage of the mixed mode is that it will secure only point-to-point as nature of Transport security.

<basicHttpBinding>

<binding name="basicHttp">

<security mode="TransportWithMessageCredential">

<Transport />

<Message clientCredentialType="UserName" />

</security>

</binding>

</basicHttpBinding>

## Both security modes

This mode both transfer security mode uses both Transport security and Message security. So message is secured using Message security and then it is transferred to the service using secure transport. This mode will maximize the security but overload the performance.

<netMsmqBinding >

<binding name="WCFSecurityExample">

<security mode="Both"/>

</binding>

</netMsmqBinding>

# SQL SERVER

## Normalization

Normalization is the process of organizing data in a database. This includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency.

First Normal Form

Eliminate repeating groups in individual tables.

Create a separate table for each set of related data.

Identify each set of related data with a primary key.

Do not use multiple fields in a single table to store similar data. For example, to track an inventory item that may come from two possible sources, an inventory record may contain fields for Vendor Code 1 and Vendor Code 2.   
  
What happens when you add a third vendor? Adding a field is not the answer; it requires program and table modifications and does not smoothly accommodate a dynamic number of vendors. Instead, place all vendor information in a separate table called Vendors, then link inventory to vendors with an item number key, or vendors to inventory with a vendor code key.

Second Normal Form

Create separate tables for sets of values that apply to multiple records.

Relate these tables with a foreign key.

Records should not depend on anything other than a table's primary key (a compound key, if necessary). For example, consider a customer's address in an accounting system. The address is needed by the Customers table, but also by the Orders, Shipping, Invoices, Accounts Receivable, and Collections tables. Instead of storing the customer's address as a separate entry in each of these tables, store it in one place, either in the Customers table or in a separate Addresses table.

Third Normal Form

Eliminate fields that do not depend on the key.

Values in a record that are not part of that record's key do not belong in the table. In general, any time the contents of a group of fields may apply to more than a single record in the table, consider placing those fields in a separate table.

## What is RDBMS?

RDBMS stands for Relational Database Management System. RDBMS data is structured in database tables, fields and records. Each RDBMS table consists of database table rows. Each database table row consists of one or more database table fields.

## What is the difference between a HAVING CLAUSE and a WHERE CLAUSE?

You can use Having Clause with the GROUP BY function in a query and WHERE Clause is applied to each row before they are part of the GROUP BY function in a query.

## What is the difference between UNION and UNION ALL SQL syntax ?

UNION SQL syntax is used to select information from two tables. But it selects only distinct records from both the table, while UNION ALL selects all records from both the tables.

## [Efficiently convert rows to columns in sql server](http://stackoverflow.com/questions/15745042/efficiently-convert-rows-to-columns-in-sql-server)

select FirstName, Amount, PostalCode, AcNumber

from

(

select Name, ColumnName

from TestTab1

) d

pivot

(

max(Name)

for ColumnName in (Firstname, Amount, PostalCode, AcNumber)

) piv;

**Column to row**

select id, entityId,

indicatorname,

indicatorvalue

from yourtable

unpivot

(

indicatorvalue

for indicatorname in (Indicator1, Indicator2, Indicator3)

) unpiv;

## Database ACID (Atomicity, Consistency, Isolation, Durability) Properties

There are a set of properties that guarantee that database transactions are processed reliably, referred to as ACID (**Atomicity, Consistency, Isolation, Durability**).

**Atomicity**

Atomicity refers to the ability of the database to guarantee that either all of the tasks of a transaction are performed or none of them are. Database modifications must follow an all or nothing rule. Each transaction is said to be atomic if when one part of the transaction fails, the entire transaction fails.

**Consistency**

The consistency property ensures that the database remains in a consistent state before the start of the transaction and after the transaction is over (whether successful or not). For example, in a storefront there is an inconsistent view of what is truly available for purchase if inventory is allowed to fall below 0, making it impossible to provide more than an intent to complete a transaction at checkout time. An example in a double-entry accounting system illustrates the concept of a true transaction. Every debit requires an associated credit. Both of these happen or neither happen.

A distributed data system is either strongly consistent or has some form of weak consistency. Once again, using the storefront example, a database needs to provide consistency and isolation, so that when one customer is reducing an item in stock and in parallel is increasing the basket by one, this is isolated from another customer who will have to wait while the data store catches up. At the other end of the spectrum is BASE (Basically Available Soft-state Eventual consistency).

Weak consistency is sometimes referred to as eventual consistency, the database eventually reaches a consistent state. Weak consistency systems are usually ones where data is replicated; the latest version is sitting somewhere in the cluster, older versions are still out there. Eventually all nodes will see the latest version.

**Isolation**

Isolation refers to the requirement that other operations cannot access or see the data in an intermediate state during a transaction. This constraint is required to maintain the performance as well as the consistency between transactions in a database. Thus, each transaction is unaware of another transactions executing concurrently in the system.

**Durability**

Durability refers to the guarantee that once the user has been notified of success, the transaction will persist, and not be undone. This means it will survive system failure, and that the database system has checked the integrity constraints and won't need to abort the transaction. Many databases implement durability by writing all transactions into a transaction log that can be played back to recreate the system state right before a failure. A transaction can only be deemed committed after it is safely in the log.

Durability does not imply a permanent state of the database. Another transaction may overwrite any changes made by the current transaction without hindering durability.

**What is difference between DELETE and TRUNCATE commands?**

Delete command removes the rows from a table based on the condition that we provide with a WHERE clause. Truncate will actually remove all the rows from a table and there will be no data in the table after we run the truncate command.

**TRUNCATE**:

TRUNCATE is faster and uses fewer system and transaction log resources than DELETE.

TRUNCATE removes the data by deallocating the data pages used to store the table's data, and only the page deallocations are recorded in the transaction log.

TRUNCATE removes all rows from a table, but the table structure, its columns, constraints, indexes and so on, remains. The counter used by an identity for new rows is reset to the seed for the column.

You cannot use TRUNCATE TABLE on a table referenced by a FOREIGN KEY constraint. Because TRUNCATE TABLE is not logged, it cannot activate a trigger.

TRUNCATE cannot be rolled back.

TRUNCATE is DDL Command.

TRUNCATE Resets identity of the table

**DELETE**:

DELETE removes rows one at a time and records an entry in the transaction log for each deleted row.

If you want to retain the identity counter, use DELETE instead. If you want to remove table definition and its data, use the DROP TABLE statement.

DELETE Can be used with or without a WHERE clause

DELETE Activates Triggers.

DELETE can be rolled back.

DELETE is DML Command.

DELETE does not reset identity of the table.

Note: DELETE and TRUNCATE both can be rolled back when surrounded by TRANSACTION if the current session is not closed. If TRUNCATE is written in Query Editor surrounded by TRANSACTION and if session is closed, it can not be rolled back but DELETE can be rolled back.

## What are the Different Normalization Forms?

## What is a Stored Procedure?

A [stored procedure](http://technet.microsoft.com/en-us/library/aa174792(v=sql.80).aspx#sql:stored_procedure) is a group of Transact-SQL statements compiled into a single execution plan.

## What is a Trigger?

A trigger is a database object that is attached to a table. In many aspects it is similar to a stored procedure. As a matter of fact, triggers are often referred to as a "special kind of stored procedure." The main difference between a trigger and a stored procedure is that the former is attached to a table and is only fired when an INSERT, UPDATE or DELETE occurs.

## What are the Different Types of Triggers?

There are two types of trigger DDL Triger and DML Trigger

## What is a View?

## What is an Index?

## What is a Linked Server?

## What is a Cursor?

## What is Collation?

## What is the Difference between a Function and a Stored Procedure?

**Stored procedure**

It is like a miniture program in SQL Server

It can be as simple as a select statement, or as complex as a long script that adds, deletes,updates, and/or reads data from multiple tables in a database.

(Storedprocedures can also implement loops and cursors which both allow you to work with smaller results or row by row operations on data.)

Should be called using EXEC or EXECUTE statement

Returns Table variables but we can't use OUT parameter

Supports transactions

**Function**

**They cannot be used to update, delete, or add records to the database.**

They simply return a single value or a table value.

They can only be use to select records. However, they can be called very easily from within standard SQL, such as: SELECT dbo.functionname('Parameter1') OR SELECT Name, dbo.Functionname('Parameter1') FROM sysObjects

For simple reusable select operations, functions can simply our code. Just be wary of using JOIN clauses in your functions. If your function has a JOIN clause and you call it from another select statement that returns multiple results, that function call with JOIN those tables together for EACH line returned in the result set. So though they can be helpful in simpling some logic, they can also be a performance bottleneck if they're not used properly

Returns the values using OUT parameter

Not supports transactions

## What is subquery? Explain the Properties of a Subquery?

## What are Different Types of Join?

## What are Primary Keys and Foreign Keys?

## What is User-defined Functions? What are the types of User-defined Functions that can be created?

## What is an Identity?

## What is DataWarehousing?

## What languages BI uses to achieve the goal?

## What is Standby Servers? Explain Types of Standby Servers.

## What is Dirty Read?

## Why can’t I use Outer Join in an Indexed View?

## What is the Correct Order of the Logical Query Processing Phases?

## What are Different Types of Locks?

## What are Pessimistic Lock and Optimistic Lock?

## When is the use of UPDATE\_STATISTICS command?

## What is the Difference between a HAVING clause and a WHERE clause?

## What is Connection Pooling and why it is Used?

## What are the Properties and Different Types of Sub-Queries?

## What are the Authentication Modes in SQL Server? How can it be Changed?

## Which Command using Query Analyzer will give you the Version of SQL Server and Operating System?

## What is an SQL Server Agent?

## Can a Stored Procedure call itself or a Recursive Stored Procedure? How many levels of SP nesting is possible?

## What is Log Shipping?

## Name 3 ways to get an Accurate Count of the Number of Records in a Table?

## What does it mean to have QUOTED\_IDENTIFIER ON? What are the Implications of having it OFF?

## What is the Difference between a Local and a Global Temporary Table?

## What is the STUFF Function and How Does it Differ from the REPLACE Function?

## What is PRIMARY KEY?

## What is UNIQUE KEY Constraint?

## What is FOREIGN KEY?

## What is CHECK Constraint?

## What is NOT NULL Constraint?

## What is the difference between UNION and UNION ALL?

## What is B-Tree?

## How to get @@ERROR and @@ROWCOUNT at the Same Time?

## What is a Scheduled Job or What is a Scheduled Task?

## What are the Advantages of Using Stored Procedures?

## What is a Table Called, if it has neither Cluster nor Non-cluster Index? What is it Used for?

## Can SQL Servers Linked to other Servers like Oracle?

## What is BCP? When is it Used?

## What Command do we Use to Rename a db, a Table and a Column?

## What are sp\_configure Commands and SET Commands?

## How to Implement One-to-One, One-to-Many and Many-to-Many Relationships while Designing Tables?

## What is Difference between Commit and Rollback when Used in Transactions?

## What is an Execution Plan? When would you Use it? How would you View the Execution Plan?

## What is Difference between Table Aliases and Column Aliases? Do they Affect Performance?

## What is the difference between CHAR and VARCHAR Datatypes?

## What is the Difference between VARCHAR and VARCHAR(MAX) Datatypes?

## What is the Difference between VARCHAR and NVARCHAR datatypes?

## Which are the Important Points to Note when Multilanguage Data is Stored in a Table?

## How to Optimize Stored Procedure Optimization?

## What is SQL Injection? How to Protect Against SQL Injection Attack?

## How to Find Out the List Schema Name and Table Name for the Database?

## What is CHECKPOINT Process in the SQL Server?

## How does Using a Separate Hard Drive for Several Database Objects Improves Performance Right Away?

## How to Find the List of Fixed Hard Drive and Free Space on Server?

## Why can there be only one Clustered Index and not more than one?

## What is Difference between Line Feed (\n) and Carriage Return (\r)?

## Is It Possible to have Clustered Index on Separate Drive From Original Table Location?

## What is a Hint?

## How to Delete Duplicate Rows?

## Why the Trigger Fires Multiple Times in Single Login?

## What is Aggregate Functions?

## What is Use of @@ SPID in SQL Server?

## What is the Difference between Index Seek vs. Index Scan?

## What is the Maximum Size per Database for SQL Server Express?

## How do We Know if Any Query is Retrieving a Large Amount of Data or very little data?

## What is the Difference between GRANT and WITH GRANT while Giving Permissions to the User?

## How to Create Primary Key with Specific Name while Creating a Table?

## What is T-SQL Script to Take Database Offline – Take Database Online

## How to Enable/Disable Indexes?

## Can we Insert Data if Clustered Index is Disabled?

## How to Recompile Stored Procedure at Run Time?

## Is there any Performance Difference between IF EXISTS (Select null from table) and IF EXISTS (Select 1 from table)?

## What is Difference in Performance between INSERT TOP (N) INTO Table and Using Top with INSERT?

## Does the Order of Columns in UPDATE statements Matter?

## What are the basic functions for master, msdb, model, tempdb and resource databases?

## What is the Maximum Number of Index per Table?

## Explain Few of the New Features of SQL Server 2008 Management Studio

## Explain IntelliSense for Query Editing

## Explain MultiServer Query

## Explain Query Editor Regions

## Explain Object Explorer Enhancements

## Explain Activity Monitors

## What is Service Broker?

## Where are SQL server Usernames and Passwords Stored in the SQL server?

## What is Policy Management?

## What is Database Mirroring?

## What are Sparse Columns?

## What does TOP Operator Do?

## What is CTE?

## What is MERGE Statement?

## What is Filtered Index?

## Which are the New Data Types Introduced in SQL SERVER 2008?

## What are the Advantages of Using CTE?

## How can we Rewrite Sub-Queries into Simple Select Statements or with Joins?

## What is CLR?

## What are Synonyms?

## What is LINQ?

## What are Isolation Levels?

## What is Use of EXCEPT Clause?

## What is XPath?

## What is NOLOCK?

## What is the Difference between Update Lock and Exclusive Lock?

## How to Copy Data from One Table to Another Table?

## What is Catalog Views?

## What is PIVOT and UNPIVOT?

## What is a Filestream?

## What is SQLCMD?

## What do you mean by TABLESAMPLE?

## What is ROW\_NUMBER()?

## What are Ranking Functions?

## What is Change Data Capture (CDC) in SQL Server 2008?

## How can I Track the Changes or Identify the Latest Insert-Update-Delete from a Table?

## What is the CPU Pressure?

## How can I Get Data from a Database on Another Server?

## What is the Bookmark Lookup and RID Lookup?

## What is Difference between ROLLBACK IMMEDIATE and WITH NO\_WAIT during ALTER DATABASE?

## What is Difference between GETDATE and SYSDATETIME in SQL Server 2008?

## How can I Check that whether Automatic Statistic Update is Enabled or not?

## How to Find Index Size for Each Index on Table?

## What is the Difference between Seek Predicate and Predicate?

## What are Basics of Policy Management?

## What are the Advantages of Policy Management?

# Window Azure

## Components of AZURE

**Windows Azure Compute** is one of the five parts of Microsoft's cloud computing services that can run various types of applications, or, in other words, we can run various types of applications using the Windows Azure Compute feature.

**Storage Service**  
  
Provides service to store data.

**Azure Fabric**  
  
This manges and monitors both of the above.

In this article, I will talk about Compute and Fabric Services.  
  
**What is Windows Azure Compute?**  
  
Windows Azure Compute is nothing but a role provider which runs the applications on Azure. Windows Azure applications can have multiple instances, each executing in its own virtual machine (VM). These VMs run 64-bit Windows Server 2008.   
  
It provides three types of role instances.

**Web Role**  
  
This role is primarily used to make it easier in creating Web-based applications. An instance of a web role has an IIS 7 configured in it. This results in an ease of creating applications using ASP.NET, WCF or other Web technologies. One can also develop applications using non-Microsoft technologies like PHP & Java where the .NET framework in not required.

**VM Role**  
  
Each instance runs a user-given Windows Server 2008 R2 image. This role can be used in moving a Windows Server application to Windows Azure while it is moving. A better way to understand this is that it is possible to provision a VM even during run time.

**Worker Role**  
  
It is used to run various Windows-based codes. Like the Web role, the Worker role does not come with pre-configured IIS, and hence, it cannot be hosted in IIS. A developer is free to use the .NET framework or other software that runs on Windows. It works like Windows services.

## Table Storage Batch Operation

A batch operation may contain up to 100 individual table operations, with the requirement that each operation entity must have same partition key. A batch with a retrieve operation cannot contain any other operations. Note that the total payload of a batch operation is limited to 4MB.

Windows Azure PortalTo run an application, a developer accesses the Windows Azure portal through her Web browser, signing in with a Windows Live ID. She then chooses whether to create a hosting account for running applications, a storage account for storing data, or both.  
  
Once the developer has a hosting account, the developer can use a Windows Azure portal to submit applications to Windows Azure. The developer has to submit configuration information along with the application which explains the platform about the number of instances to run for each role. Windows Azure's fabric controller creates a VM for each instance and runs the code for the appropriate role in VM.  
  
Higher level walks on Users interaction with Azure.  
  
When the user sends a request to an application hosted on Azure (that can be passed by protocols like HTTP, HTTPS or TCP), the request will be received from the load balancer. The Load balancer balances the load across all the instances of the role (Web, Worker, and VM).  
  
A developer can use any combination of VM Role, Web Role or Worker Role to create an application. Based on the load of the application, the developer can use the Azure Portal and request more instances for any of the roles used in the application and vice versa if the load decreases using the API exposed by Azure.  
  
Azure Fabric  
  
Now, let us move to little depth on Azure. What is the main core of Azure?  
  
The Azure fabric is the main core concept over here. It provides a service called the Azure Fabric Controller. It is called as OS for the Azure. Because it handles/manages:

* All roles (computing) and resources.
* Deployment and activating services.
* Health monitoring for all services.
* Allocating, releasing of resources.
* Provisioning VM, terminating etc.
* Updating patches for installed OS on VM automatically. So there would be better to have two instance of roles and also no need to worry about software updates for user.

How does the Azure fabric communicate with computing and storage services?  
  
Yes. It will communicate with compute and storage services using the Azure agent. The Azure agent will reside in the roles VM. Also it exposes a Windows Azure-maintained log, sending alerts to its owner via the Azure fabric.

**What is Blob Storage?**

Azure Blob storage is a service for storing large amounts of unstructured data that can be accessed from anywhere in the world via HTTP or HTTPS. A single blob can be hundreds of gigabytes in size.

**Storage Account:** All access to Azure Storage is done through a storage account. See [Azure Storage Scalability and Performance Targets](http://msdn.microsoft.com/en-us/library/dn249410.aspx) for details about storage account capacity.

**Container:** A container provides a grouping of a set of blobs. All blobs must be in a container. An account can contain an unlimited number of containers. A container can store an unlimited number of blobs.

**Blob:** A file of any type and size. There are two types of blobs that can be stored in Azure Storage: block and page blobs. Most files are block blobs. A single block blob can be up to 200 GB in size. This tutorial uses block blobs. Page blobs, another blob type, can be up to 1 TB in size, and are more efficient when ranges of bytes in a file are modified frequently. For more information about blobs, see [Understanding Block Blobs and Page Blobs](http://msdn.microsoft.com/en-us/library/windowsazure/ee691964.aspx).

**Block Blobs and Page Blobs**

Block blobs let you upload large blobs efficiently. Block blobs are comprised of blocks, each of which is identified by a block ID. You create or modify a block blob by writing a set of blocks and committing them by their block IDs. Each block can be a different size, up to a maximum of 4 MB. The maximum size for a block blob is **200 GB**, and a block blob can include no more than 50,000 blocks.

Page blobs are a collection of 512-byte pages optimized for random read and write operations. To create a page blob, you initialize the page blob and specify the maximum size the page blob will grow. To add or update the contents of a page blob, you write a page or pages by specifying an offset and a range that align to 512-byte page boundaries. A write to a page blob can overwrite just one page, some pages, or up to 4 MB of the page blob. Writes to page blobs happen in-place and are immediately committed to the blob. The maximum size for a page blob is 1 TB.

<http://azure.microsoft.com/en-us/documentation/articles/storage-dotnet-how-to-use-blobs/>

**URL format:** Blobs are addressable using the following URL format:  
http://<storage account>.blob.core.windows.net/<container>/<blob>

The following example URL could be used to address one of the blobs in the diagram above:  
<http://sally.blob.core.windows.net/movies/MOV1.AVI>

***Blob Snapshots***

A snapshot is a read-only version of a blob stored as it was at the time the snapshot was created. You can use snapshots to create a backup or checkpoint of a blob. A snapshot blob name includes the base blob URI plus a date-time value that indicates when the snapshot was created.

**What is the Table Service?**

The Azure Table storage service stores large amounts of structured data. The service is a NoSQL datastore which accepts authenticated calls from inside and outside the Azure cloud. Azure tables are ideal for storing structured, non-relational data. Common uses of the Table service include:

Storing TBs of structured data capable of serving web scale applications

Storing datasets that don't require complex joins, foreign keys, or stored procedures and can be denormalized for fast access

Quickly querying data using a clustered index

Accessing data using the OData protocol and LINQ queries with WCF Data Service .NET Libraries

**Table**: A table is a collection of entities. Tables don't enforce a schema on entities, which means a single table can contain entities that have different sets of properties. The number of tables that a storage account can contain is limited only by the storage account capacity limit.

**Entity**: An entity is a set of properties, similar to a database row. An entity can be up to 1MB in size.

**Properties**: A property is a name-value pair. Each entity can include up to 252 properties to store data. Each entity also has 3 system properties that specify a partition key, a row key, and a timestamp. Entities with the same partition key can be queried more quickly, and inserted/updated in atomic operations. An entity's row key is its unique identifier within a partition.

## What is Windows Azure?

Ans. (First Answer)Windows Azure is Microsoft Cloud Computing Operating System. Someone needs to worry about the hardware requirement, network requirement and the correct Operating System and much other stuff when he starts designing an application. Windows Azure will help the developer to get rid of these things and let him concentrate on main area which is nothing but automation and business needs.

Windows Azure is a cloud services operating system that serves as the development, service hosting and service management environment for the Windows Azure platform. Windows Azure provides developers with on-demand compute and storage to host, scale, and manage web applications on the internet through Microsoft datacenters.

**What is Windows Azure Platform?**

A collective name of Microsoft’s Platform as a Service (PaaS) offering which provides a programming platform, a deployment vehicle, and a runtime environment of cloud computing hosted in Microsoft datacenters.

**What is difference between Windows Azure Platform and Windows Azure?**The former is Microsoft’s PaaS offering including Windows Azure, SQL Azure, and Appfabric; while the latter is part of the offering and the Microsoft’s cloud OS.

**What is fabric?**  
In the Windows Azure cloud fabric is nothing but a combination of many virtualized instances which run client application.

**What are the three core components of Windows Azure?**Windows Azure has three core components: Compute, Storage and Fabric.

Compute: It provides a computation environment with Web Role, Worker Role, and VM Role.

Storage: It focuses on providing scalable storage (Blobs, non-relational Tables, and Queues) for large-scale needs. Relational Database functionality is offered through SQL Azure, which is a scalable version of SQL Server that runs on the Azure platform.

Fabric: Fabric (Windows Azure Fabric) makes up the physical underpinnings of the Windows Azure platform as the network of interconnected nodes consisting of servers, high-speed connections, and switches. Conceptually, the repetitive pattern of nodes and connections suggests a woven or fabric-like nature. Compute and Storage components are part of the Fabric.

**What is the downtime for applications in case of any patching?  
Ans.** Windows Azure will have replicas for each and every application and provide zero downtime in case of any security patching.

**How many copies of data are maintained in Windows Azure?  
Ans.** Windows Azure provides you with three copies of data. This makes your application running on very reliable data.

**What is queue storage in the Windows Azure?  
Ans.** Queue storage gives you capability of sending the small data just as messages. Queue storage basically helps informing the task to the worker threads.

**What is Windows Azure AppFabric? What are the services it provides?  
Ans.** Windows Azure AppFabric is a part of the Microsoft Windows Azure Platform. Positioned by Microsoft as middleware, it provides several cloud computing services:

**Access Control Service:** Provides an easy way for web applications and web services to identify users, and grant access control to the applications and services.

**AppFabric Applications:** Provides tools and APIs for developing and hosting a composite application expressly for deployment through the service.

**Service Bus:** Provides secure connections between distributed and disconnected applications in the cloud.

**Caching:** Provides a distributed, in-memory, application cache service for Windows Azure and SQL Azure applications.

**Integration:** Leverages several capabilities of Microsoft BizTalk Server such as transforming messages from one format to another, or providing a queue for message passing.

**What are the main uses of Windows Azure?  
Ans.** 1. Run enterprise workloads in the cloud  
2. Build, modify, and distribute scalable applications with minimal on-premises resources  
3. Perform large-volume storage, batch processing, intense or large-volume computations  
4. Create, test, debug, and distribute Web services quickly and inexpensively

What are benefits of Windows Azure?Bring your ideas to market faster and pay as you go  
2. Reduce costs of building and extending on-premises resources  
3. Reduce the effort and costs of IT management  
4. Respond quickly to changes in your business and customer needs  
5. Choose an on-premises or off-premises deployment model that best suits your needs.  
6. Scale your IT resources up and down based on your needs.  
7. Consume computing resources ONLY when the needs arise.  
8. Focus less energy on managing operational resources and constraints.  
9. Remove the need to manage hardware  
10. Use your existing development skills to build cloud applications  
11. Consistent development and management experience across on-premises and the cloud.

What is worker role? A role that provides a general-purpose environment for running application workloads. Typically, background processing for web applications running on web roles can be offloaded to instances of the worker role.

What is web role?Ans. A role that provides an environment for running web sites or applications as supported by Internet Information Services (IIS) 7.0.

What do you mean by VM role?Ans. A role that provides a customer-customizable virtual machine in which to run the customer’s hosted service.

What is Virtual IP(VIP)?An IP address that is not connected to a specific computer or network interface card. Service deployments are assigned a VIP for receiving network traffic which is redirected to a physical device in the Windows Azure fabric.

What is use of activation token? A unique identifier that activates a Windows Azure Connect endpoint and is linked to the customer’s subscription. The token can be reset by the customer.

What do you mean by affinity group?Ans. A geographical grouping of a customer’s hosted service deployments and storage accounts within Windows Azure. An affinity group can improve service performance by locating compute and data storage workloads in the same data center or near the target user audience.

What is cloud? A collection of servers, networks, and gateways that provide a computing environment that is delivered over the Internet as a service on a pay-per-use basis.

What do you mean by Compute Emulator and Compute hours? **Compute Emulator:** The software emulation of the Windows Azure Compute service that runs in the developer’s local environment for application testing and debugging.

**Compute Hours:** The amount of time charged to a subscription for hosted services deployed and running in the Windows Azure compute environment. Customers are charged different rates for different sized compute instances.

What is a container?Ans. It is a user-defined set of blobs within a storage account. A container resource has no associated content, only properties and metadata.

What is cspack?Ans. It is a command-line tool that generates a service package file (.cspkg) and prepares an application for deployment, either to Windows Azure or to the compute emulator.

What is csrun? It is a command-line tool that deploys a packaged application to the Windows Azure compute emulator and manages the running service.

What is guest OS? It is the operating system that runs on the virtual machine that hosts an instance of a role.

What is input endpoint?  
 It is the IP and port on which a role instance receives inbound traffic. Each type of role has restrictions on the number and type of input endpoints that can be defined. A web role can have no more than one HTTP endpoint and one HTTPS endpoint. A work role can have no more than five HTTP, HTTPS, or TCP endpoints (any combination). A VM role can have only one HTTP, HTTPS, or TCP endpoint.

What do you mean by page blob? It is a collection of pages in a blob. A page is a range of data that is identified by its offset from the start of the blob.

***What is a Fabric Agent?***

A self-initialized application deployed with the root partition of a Windows Azure Compute node to form the fabric.

What is Content Delivery Network (CDN)?An add-on feature to Windows Azure subscription to cache Windows Azure BLOBs and the static content output of Compute instances at Microsoft’s caching servers near what the content is most frequently accessed.  
 ***What is SQL Azure?***A cloud-based relational database service with SQL Azure Reporting.

***What is the use of Window Azure SDK?***With Windows Azure SDK, a Visual Studio programmer can employ Cloud project template, develop Windows Azure applications, and securely publish an application to cloud using certificate.

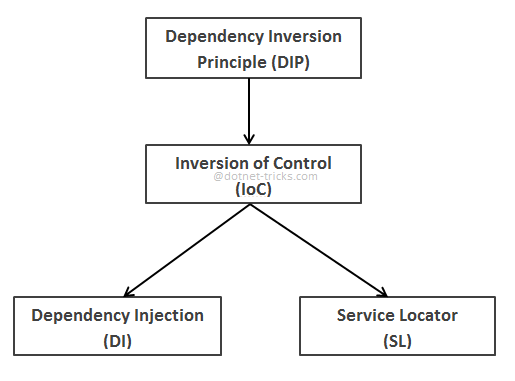
# Design Pattern

## Dependency Injector

A DI Container is a framework to create dependencies and inject them automatically when required. It automatically creates objects based on request and inject them when required. DI Container helps us to manage dependencies with in the application in a simple and easy way.

We can also manage an application dependencies without a DI Container, but it will be like as POOR MAN’S DI and we have to do more work, to make it configured and manageable.

## Understanding Inversion of Control, Dependency Injection and Service Locator



More over IoC is a generic term and it is not limited to DI. Actually, DI and Service Locator patterns are specialized versions of the IoC pattern or you can say DI and Service Locator are the ways of implementing IoC.

**For example**, Suppose your Client class needs to use a Service class component, then the best you can do is to make your Client class aware of an IService interface rather than a Service class. In this way, you can change the implementation of the Service class at any time (and for how many times you want) without breaking the host code.

Ref: <http://satoricode.net/2011/10/01/UnderstandingTheBenefitsOfADependencyInjectionContainerIOC.aspx>

## IoC and DI

The terms Dependency Injection (DI) & Inversion of Control (IoC) are generally used interchangeably to describe the same design pattern. The pattern was originally called IoC, but Martin Fowler proposed the shift to DI because all frameworks invert control in some way and he wanted to be more specific about which aspect of control was being inverted.

## Dependency Injection (DI)

DI is a software design pattern that allow us to develop loosely coupled code. DI is a great way to reduce tight coupling between software components. DI also enables us to better manage future changes and other complexity in our software. The purpose of DI is to make code maintainable.

The Dependency Injection pattern uses a builder object to initialize objects and provide the required dependencies to the object means it allows you to "inject" a dependency from outside the class.

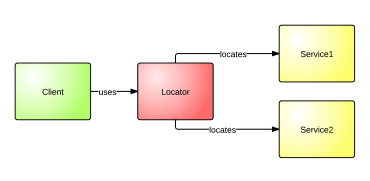
## Dependency Injection VS Service Locator

When you use a service locator, every class will have a dependency on your service locator. This is not the case with dependency injection. The dependency injector will typically be called only once at startup, to inject dependencies into the main class.

The Service Locator pattern is easier to use in an existing codebase as it makes the overall design looser without forcing changes to the public interface. Code that is based on the Service Locator pattern is less readable than the equivalent code that is based on Dependency Injection.

Dependency Injection is an implementation of the Inversion of Control pattern. There are two possible implementations for IoC:

**Service Locator or Dependency Lookup**: container provides callback functionality and lookup context, components are requested by using the locator (container) API. A dependency to the locator and API persists, central binding and contextual binding is possible.



**Dependency Injection**: implementation without dependency to a container API possible, able to manage lifecycles (scope, request, singleton, thread, transient), lookup and contextual binding abilities are depending on the DI framework.

## Simple Dependency Injection

Let me provide you an example of DI without using any DI framework. It’s a common approach, that you likely know.

When an object needs another object to operate properly, we have a dependency. In the following example we have a **Service** and a **Client**, where the **Client** class is tightly coupled with the **Service** class:

|  |  |
| --- | --- |
|  | public class Service  {  public void Serve()  {  // do some serving stuff  }}  public class Client  {  private Service \_service;  public Client()  {  this.\_service = new Service();  }  public void Start()  {  this.\_service.Serve();  // do some client stuff  } |

Dependency injection eliminates that tight coupling and makes the application more flexible and reusable:

public interface IService

{

public void Serve();

}

public class Service : IService

{

public void Serve()

{

// do some serving stuff

}

}

public class Client

{

private IService \_service;

public Client(IService service)

{

this.\_service = service;

}

public void Start()

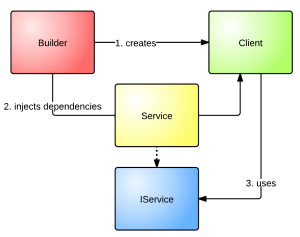
{

this.\_service.Serve();

// do some client stuff

}

}



Sample Builder-Implementation:

|  |  |
| --- | --- |
|  | var client = new Client(new Service());  client.Start(); |

## Service Locator

A common approach to achieve assembling the **Client** could be the following implementation, by using service locator to avoid some of the issues described before:

public class Locator

{

private static Dictionary<Type, Type> dictionary = new Dictionary<Type, Type>();

static Locator()

{

dictionary.Add(typeof(IService), typeof(Service));

}

public object Create(Type type)

{

if ((type == null || !dictionary.ContainsKey(type)))

throw new NullReferenceException();

return Locator.CreateInstance(dictionary[type]);

}

public T Create<T>()

{

return (T)Create(typeof(T));

}

}

Usage:

|  |  |
| --- | --- |
|  | **var client = new Client(Locator.Create<IService>());** |

## Repository pattern

A repository basically works as a mediator between our business logic layer and our data access layer of the application. Sometimes, it would be troublesome to expose the data access mechanism directly to business logic layer, it may result in redundant code for accessing data for similar entities or it may result in a code that is hard to test or understand. To overcome these kinds of issues, and to write an Interface driven and test driven code to access data, we use Repository Pattern. The repository makes queries to the data source for the data, thereafter maps the data from the data source to a business entity/domain object, finally and persists the changes in the business entity to the data source.

It centralizes the data logic or Web service access logic.

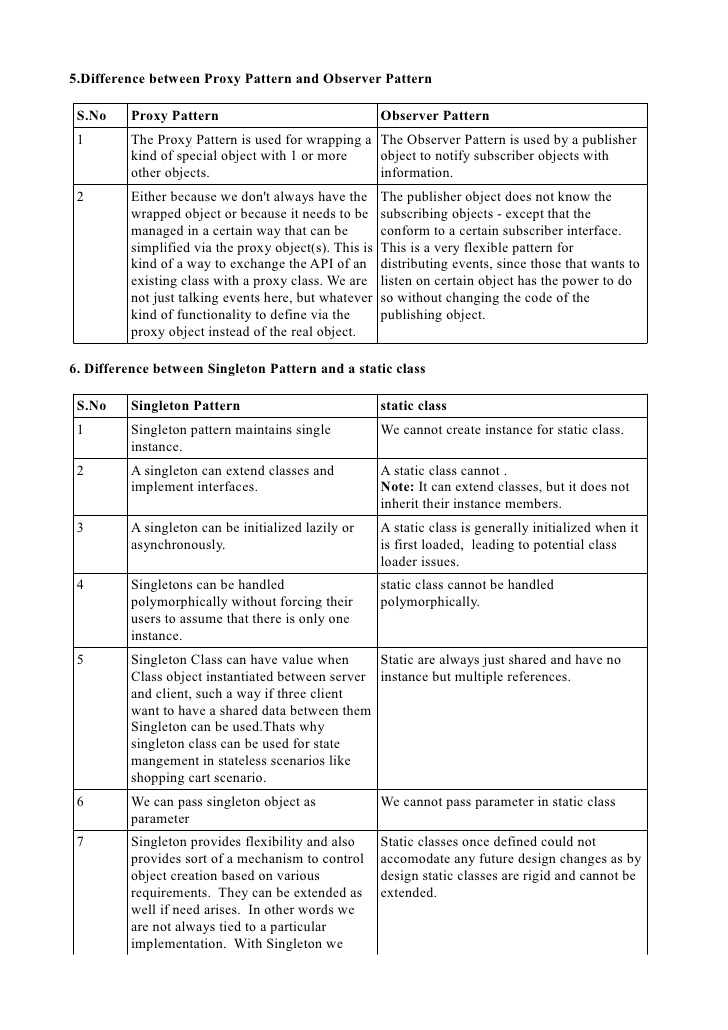
It provides a substitution point for the unit tests.

It provides a flexible architecture that can be adapted as the overall design of the application evolves.

When we use Entity Framework, as we did in our last application created, we were calling the Entity Framework class object in the controller class for accessing the entity classes. Now we can say that that system was somewhat a tightly coupled system. To overcome this situation, as we discussed, we’ll implement Repository Pattern.

In Repository, we write our whole business logic of CRUD operations with the help of Entity Framework classes, that will not only result in meaningful test driven code but will also reduce our controller code of accessing data.

## [*What is Inversion of Control?*](http://stackoverflow.com/questions/3058/what-is-inversion-of-control)

* Creational Patterns
* 

[**Abstract Factory Pattern**](https://csharpdesignpatterns.codeplex.com/wikipage?title=Abstract%20Factory%20Pattern&referringTitle=Home)**:**

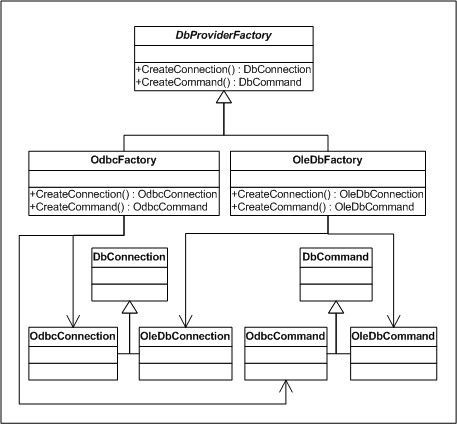
* Create a set of related objects, or dependent objects which must be used together.
* System should be configured to work with multiple families of products.
* The creation of objects should be independent from the utilizing system.
* Concrete classes should be decoupled from clients.
* Internally, Abstract Factory use Factory design pattern for creating objects. But it can also use Builder design pattern and prototype design pattern for creating objects. It completely depends upon your implementation for creating objects.
* Abstract Factory can be used as an alternative to Facade to hide platform-specific classes.
* When Abstract Factory, Builder, and Prototype define a factory for creating the objects, we should consider the following points :
* Abstract Factory use the factory for creating objects of several classes.
* Builder use the factory for creating a complex object by using simple objects and a step by step approach.
* Prototype use the factory for building a object by copying an existing object.

Create instances of classes belonging to different families

Provide an interface for create families of related dependent objects without specifying their concrete class.

Example : DBFactory object has dbconnection and dbcommand. We have sqlconnectionfactory, OracleConnectionFactory both implement dbProviderFactory

  System.Data.Common.DbProviderFactory  
    [System.Data.EntityClient.EntityProviderFactory](http://msdn.microsoft.com/en-us/library/system.data.entityclient.entityproviderfactory(v=vs.110).aspx)  
    [System.Data.Odbc.OdbcFactory](http://msdn.microsoft.com/en-us/library/system.data.odbc.odbcfactory(v=vs.110).aspx)  
    [System.Data.OleDb.OleDbFactory](http://msdn.microsoft.com/en-us/library/system.data.oledb.oledbfactory(v=vs.110).aspx)  
    [System.Data.OracleClient.OracleClientFactory](http://msdn.microsoft.com/en-us/library/system.data.oracleclient.oracleclientfactory(v=vs.110).aspx)  
    [System.Data.SqlClient.SqlClientFactory](http://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqlclientfactory(v=vs.110).aspx)



|  |  |
| --- | --- |
|  |  |

First of all, I would suggest you to read about the Abstract Factory pattern, for example [here](http://en.wikipedia.org/wiki/Abstract_factory_pattern). Now I will try to explain why you would use this pattern.

Normally, if you use the Factory pattern, you will create objects in a Factory. The problem arises when you have multiple implementation of a given class (or classes). Now, those multiple implementations are grouped. You will use the Abstract Factory pattern when you have a factory, but you would like to group the creating of objects per group.

Okay, above explanation might not be completely clear, so I will give you an example.

Let's say you have a class library with data agents. Data agents provide you methods to access and store different data. Of course, there are multiple ways of storing your data. For example: in a database, in XML file, over a service, . For each of these possible ways, you would like to have data agents. Now the problem is, you don't want that someone uses the DataAgentA for XML files together with DataAgentB for database (let's assume that we have entities A and B). The user should use only one storage engine.

Let me introduce you the Abstract Factory pattern.

You will make sure that users cannot directly instantiate your Data Agents, but they will have to get these data agents out of a factory. (An extra advantage is, that when you use for example a database (EF), you can do internal wiring to make sure your Data Agents use the same context, etc.) How do we accomplish this? We set the constructor of our data agents to ´internal´. Apart from that, we create different factories for each storage engine. Now, since those factories all do the same, we also have these interfaced (just like our data agents, since they all have to do the same, right!?).

Below we have our interfaces. Basically this is the factory pattern, but only now instead of about *classes*, we are talking about *interfaces*.

public interface IAgentA

{

// Add some methods here!

}

public interface IAgentB

{

// Add some methods here!

}

public interface IAgentFactory

{

IAgentA CreateAgentA();

IAgentB CreateAgentB();

}

Now for the two agents, we have two possible implementations, one for XML and one for database storage (again: this is an example, you can have as many implementation types as you want). Those implementations would look like this (see below). Please note that I made the constructor internal! This is needed for the part that comes after this code block.

public class AgentA\_Xml : IAgentA

{

internal AgentA\_Xml()

{ /\* Construction here \*/}

// IAgentA method implementations

}

public class AgentB\_Xml : IAgentB

{

internal AgentB\_Xml()

{ /\* Construction here \*/}

// IAgentB method implementations

}

public class AgentA\_Database : IAgentA

{

internal AgentA\_Database()

{ /\* Construction here \*/}

// IAgentA method implementations

}

public class AgentB\_Database : IAgentB

{

internal AgentB\_Database()

{ /\* Construction here \*/}

// IAgentB method implementations

}

Now as the constructors are internal. This causes that you cannot instantiate those classes outside the assembly, which is generally what you do with these kinds of cases. Now we have to create our factories.

public class XMLAgentFactory : IAgentFactory

{

public IAgentA CreateAgentA()

{

return new AgentA\_Xml();

}

public IAgentB CreateAgentB()

{

return new AgentB\_Xml();

}

}

public class DatabaseAgentFactory : IAgentFactory

{

public IAgentA CreateAgentA()

{

return new AgentA\_Database();

}

public IAgentB CreateAgentB()

{

return new AgentB\_Database();

}

}

Since both factories implement the IAgentFactory interface, the user can easily change of AgentFactory implementation (if he, in this case, wants to use a different storage engine) without having to change any other code he wrote (against the agents), as long as he programmed against the interfaces (obviously).

Above explanation hopefully answers your questions (1) and (2).

Good example for Abstract factory pattern in C#?

and what are advantages of Abstract factory pattern in c#?

Answering your question (3).

How use C# generics with Abstract factory pattern?

You can still use generics, this doesn't change any bit when you use an Abstract Factory pattern. Of course, you will have to create generic factory methods (the create methods), but that shouldn't be any problem.

[**Builder Pattern**](https://csharpdesignpatterns.codeplex.com/wikipage?title=Builder%20Pattern&referringTitle=Home)**:** Separate representation and object construction

[**Factory Method Pattern**](https://csharpdesignpatterns.codeplex.com/wikipage?title=Factory%20Method%20Pattern&referringTitle=Home)**:**

Define an interface for creating an object, but let subclasses decide which class to instantiate. Factory Method lets a class defer instantiation to subclasses.

Create instances of derived classes

Define an interface for creating an object, but let subclass decide which class to instantiate. Factory method let a class differ instantiation to subclasses.

This strategy for creating new object instances known as a factory pattern. Rather than invoking the object constructor you can as the object factory to create a instance for you. The way the factory class can hide the complexity of object creation (like how to parse a Double out of string). If you can wanted to change the details of creating object you would only have to change the factory itself; you would not have to change every single place if the code where constructor is call.

Example:

Convert.ToString()

Convert.ToDateTime()

[**Prototype Pattern**](https://csharpdesignpatterns.codeplex.com/wikipage?title=Prototype%20Pattern&referringTitle=Home) **Design Pattern**

Clone or copy initialized instances

[**Singleton Pattern**](https://csharpdesignpatterns.codeplex.com/wikipage?title=Singleton%20Pattern&referringTitle=Home) **Design Pattern**

Class with only one single possible instance

Singleton pattern that I've seen is SQL Connection Pooling in .NET

Step 1: we are making any outside class cannot create an instance; to do this, we need private or protected constructor.

Step2: create a public static method to access this class instance. in this public static method give an instance your class for a single time(create instance of class as a static member variable).

public class SingletonDemo{

private static SingletonDemo instance=null;

private SingltonDemo(){}

public static SingletonDemo getAnInstance(){

if(instance==null)

instance=new SingletonDemo();

return instance;

}

}

ServiceHost.SingletonInstance example for singleton

[**Difference between static class and singleton pattern?**](http://stackoverflow.com/questions/519520/difference-between-static-class-and-singleton-pattern)

1. Singleton object stores in **Heap** but, static object stores in **stack**
2. We can clone the object of Singleton but, we cannot clone the static class object
3. Singleton class follow the OOP(object oriented principles) but not static class
4. we can implement interface with Singleton class but not with Static class

# Structural Patterns

## [Adapter Pattern](https://csharpdesignpatterns.codeplex.com/wikipage?title=Adapter%20Pattern&referringTitle=Home):

Match interfaces of classes with different interfaces

Abstract factory pattern in useful when the client needs to create objects which are somehow related. If we need to create the family of related or dependent objects, then we can use Abstract Factory Pattern.

This pattern is particularly useful when the client doesn't know exactly what type to create. As an example, let's say a Showroom exclusively selling cellphones gets a query for the smart phones made by Samsung. Here we don't know the exact type of object to be created (assuming all the information for a phone is wrapped in the form of a concrete object). But we do know that we are looking for smart phones that are manufactured by Samsung. This information can actually be utilized if our design has Abstract factory implementation

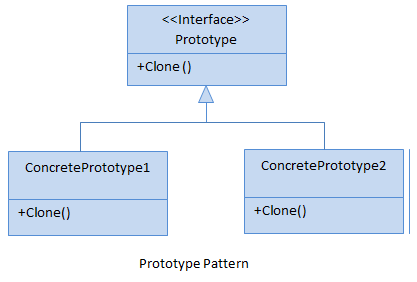
## What is Prototype Pattern?

Prototype pattern is used to create a duplicate object or clone of the current object to enhance performance. This pattern is used when creation of object is costly or complex.

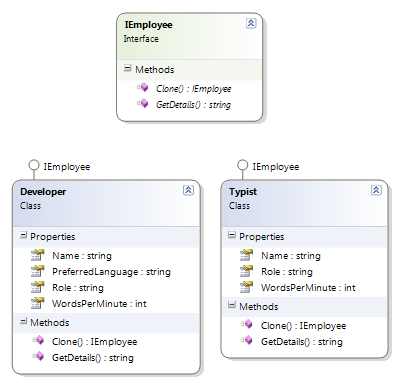
For Example: An object is to be created after a costly database operation. We can cache the object, returns its clone on next request and update the database as and when needed thus reducing database calls.

**Prototype Pattern - UML Diagram & Implementation**

The UML class diagram for the implementation of the Prototype design pattern is given below:



The classes, interfaces and objects in the above UML class diagram are as follows:



**Who is what?**

The classes, interfaces and objects in the above class diagram can be identified as follows:

**IEmployee** - Prototype interface

Developer & Typist- Concrete Prototype

C# - Sample Code

*/// <summary>*

*/// The 'Prototype' interface*

*/// </summary>*

public interface IEmployee

{

IEmployee Clone();

string GetDetails();

}

*/// <summary>*

*/// A 'ConcretePrototype' class*

*/// </summary>*

public class Developer : IEmployee

{

public int WordsPerMinute { get; set; }

public string Name { get; set; }

public string Role { get; set; }

public string PreferredLanguage { get; set; }

public IEmployee Clone()

{

*// Shallow Copy: only top-level objects are duplicated*

return (IEmployee)MemberwiseClone();

*// Deep Copy: all objects are duplicated*

*//return (IEmployee)this.Clone();*

}

public string GetDetails()

{

return string.Format("{0} - {1} - {2}", Name, Role, PreferredLanguage);

}

}

*/// <summary>*

*/// A 'ConcretePrototype' class*

*/// </summary>*

public class Typist : IEmployee

{

public int WordsPerMinute { get; set; }

public string Name { get; set; }

public string Role { get; set; }

public IEmployee Clone()

{

*// Shallow Copy: only top-level objects are duplicated*

return (IEmployee)MemberwiseClone();

*// Deep Copy: all objects are duplicated*

*//return (IEmployee)this.Clone();*

}

public string GetDetails()

{

return string.Format("{0} - {1} - {2}wpm", Name, Role, WordsPerMinute);

}

}

*/// <summary>*

*/// Prototype Pattern Demo*

*/// </summary>*

class Program

{

static void Main(string[] args)

{

Developer dev = new Developer();

dev.Name = "Rahul";

dev.Role = "Team Leader";

dev.PreferredLanguage = "C#";

Developer devCopy = (Developer)dev.Clone();

devCopy.Name = "Arif"; *//Not mention Role and PreferredLanguage, it will copy above*

Console.WriteLine(dev.GetDetails());

Console.WriteLine(devCopy.GetDetails());

Typist typist = new Typist();

typist.Name = "Monu";

typist.Role = "Typist";

typist.WordsPerMinute = 120;

Typist typistCopy = (Typist)typist.Clone();

typistCopy.Name = "Sahil";

typistCopy.WordsPerMinute = 115; *//Not mention Role, it will copy above*

Console.WriteLine(typist.GetDetails());

Console.WriteLine(typistCopy.GetDetails());

Console.ReadKey();

}

}

Prototype Pattern Demo - Output

http://www.dotnet-tricks.com/Content/images/designpatterns/Prototype2.png

**When to use it?**

The creation of each object is costly or complex.

A limited number of state combinations exist in an object.

* **[Bridge Pattern](https://csharpdesignpatterns.codeplex.com/wikipage?title=Bridge%20Pattern&referringTitle=Home)::** Separate implementation and object interfaces
* **Composite:** Simple and composite objects tree
* **Decorator:** Dynamically add responsibilities to objects
* **Facade:** Class that represents subclasses and subsystems
* **Flyweight:** Minimize memory usage by sharing as much data as possible with similar objects
* **Proxy:** Object that represents another object

# Behavioral Patterns

## Chain of Responsibility:

* Pass requests between command and processing objects within a chain of objects

## Command:

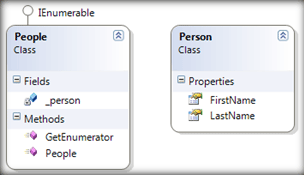
* Encapsulate a method call as an object containing all necessary information

## Interpreter:

* Include language elements and evaluate sentences in a given language

## Iterator pattern :

Give sequential access to elements in a collection

The Iterator pattern provides a way of accessing elements of a collection sequentially, without knowing how the collection is structured. As an extension, the pattern allows for filtering elements in a variety of ways as they are generated.  
  
Provide a way to access the elements of an aggregate object sequentially without exposing its underlying representation.  
  
**Design Pattern :**  
  
The concept of iterators and enumerators (also called generators) has been around for a long time. Enumerators are responsible for producing the next element in a sequence defined by certain criteria. Such a sequence is said to be enumerable. Examples might be the next prime number or the next product order sorted by date. The iterator is the means by which we cycle through this sequence of elements from beginning to end.  
  
**UML Diagram**  
  
  
  
**C# support for the pattern**    public class Person  
    {  
        public string FirstName { get; set; }  
        public string LastName { get; set; }  
    }

* public class People : IEnumerable  
      {  
          private Person[] \_person;
* public People(Person[] person)  
          {   
              \_person = new Person[person.Length];
* for (int i = 0; i < person.Length; i++)  
                  \_person[i] = person[i];  
          }
* public IEnumerator GetEnumerator()  
          {  
              foreach (Person p in \_person)  
                  **yield return p; //Yield return value to the Enumerator object.**        }  
      }
  + static void Main(string[] args)  
          {  
                Person[] peopleArray = new Person[3]  
                    {  
                new Person(){ FirstName= "Chinna",  LastName = "Srihari"},  
                new Person(){ FirstName= "Chinna",  LastName="Sushma"},  
                new Person() { FirstName= "Chinna", LastName= "Lohetha"},  
                };
* People p = new People(peopleArray);
* foreach (Person p1 in p)  
                  Console.WriteLine(p1.FirstName + ", " + p1.LastName);  
   }

## Mediator Pattern:

Encapsulates and simplifies communication between objects

## Memento Pattern :

Undo modifications and restore an object to its initial state

## Observer Pattern :

* Notify dependent objects of state changes

## State Pattern:

Change object behavior depending on its state

The State Pattern (also referred to as State Design Pattern) is one of the behavioral patterns. It defines a manner for controlling communication among classes or entities. It is used to change the behavior of the object when the object's internal state is changed. This pattern helps objects to change it state without changing the interface (used to access the object) or lose the current state. This change of the state of the class or object is hidden from the outer world with the use of a context (also known as wrapper object). The State Pattern can be used to implement complex decision-making algorithms represented as a flow chart or state diagram. A State Pattern can also be implement where various states exist and each state has a specific action.

 public interface StateBase

{

    void Change(Context context);

}

public class StateA : StateBase

{

    public void Change(Context context)

    {

        //Change state of context from A to B.

        context.State = new StateB();

        Console.WriteLine("Change state from A to B.");

    }

}

public class StateB : StateBase

{

    public void Change(Context context)

    {

        //Change state of context from B to C.

        context.State = new StateC();

        Console.WriteLine("Change state from B to C.");

    }

}

public class StateC : StateBase

{

    public void Change(Context context)

    {

        //Change state of context from C to A.

        context.State = new StateA();

        Console.WriteLine("Change state from C to A.");

    }

}

**Client Code**

public class Context

{

    public Context(StateBase state)

    {

        State = state;

        Console.WriteLine("Create object of context class with initial State.");

    }

    public StateBase State { get; set; }

    /// <summary>

    /// State change request

    /// </summary>

    public void Request()

    {

        State.Change(this);

    }

}  
   
**Sample code and output**

static void Main(string[] args)

{

    // create Context object and suplied current state or initial state (state A).

    Context context = new Context(new StateA());

    //Change state request from A to B.

    context.Request();

    //Change state request from B to C.

    context.Request();

    //Change state request from C to A.

    context.Request();

}

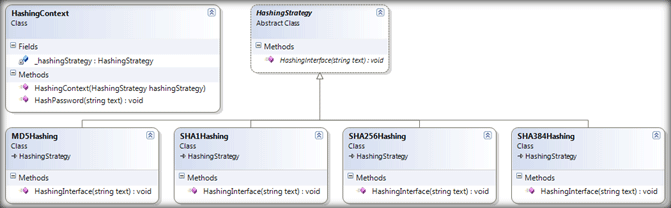
* Strategy Design Pattern

The Strategy pattern involves separating an algorithm from its host class and putting it in a separate class. When there are multiple strategies available for an algorithm, for a given problem it is always better to separate them in to different objects. If the algorithms are all kept in the one class, the class will be become a big messy conditional statements.  
  
The Strategy pattern enables a client to choose which algorithm to use from a family of algorithms and gives it a simple way to access it.   
  
Define a family of algorithms, encapsulate each one, and make them interchangeable. Strategy lets the algorithm vary independently from the clients that use it.  
  
Note:

Strategy is like Template Method except the Strategy patter consists of details whereas template provides a skeleton for the algorithm.

State is like Strategy except in its objective.

Strategy lets us change the internal object. Decorator lets us change the covering, underlying object remains same.

Design  
  
Let us say that we are implementing text hashing for the system. During the requirements definition the decision is to implement the MD5 hashing; during development SHA1 is added for another requirement, and it kept continuing. Finally the decision is to implement with a couple of options and based on the client strategy it can be changed.  
  
UML Diagram   
  
  
Code  
  
    public abstract class HashingStrategy  
    {  
        public abstract void HashingInterface(string text);  
    }

    public class MD5Hashing : HashingStrategy  
    {  
        public override void HashingInterface(string text)  
        {  
            MD5 md5 = new MD5CryptoServiceProvider();  
  
            Byte[] bytes;  
            bytes = ASCIIEncoding.Default.GetBytes(text);

            Byte[] encodedBytes;  
            encodedBytes = md5.ComputeHash(bytes);

            string A = BitConverter.ToString(encodedBytes);  
        }  
    }

    public class HashingContext

    {  
        private HashingStrategy \_hashingStrategy;

        public HashingContext(HashingStrategy hashingStrategy)  
        {  
            \_hashingStrategy = hashingStrategy;  
        }

        public void HashPassword(string text)  
        {  
            \_hashingStrategy.HashingInterface(text);  
        }  
    }

    public class SHA1Hashing : HashingStrategy  
    {  
        public override void HashingInterface(string text)  
        {  
            SHA1 sha1 = new SHA1CryptoServiceProvider();

            Byte[] bytes;  
            bytes = ASCIIEncoding.Default.GetBytes(text);

            Byte[] encodedBytes;  
            encodedBytes = sha1.ComputeHash(bytes);

            string A = BitConverter.ToString(encodedBytes);  
        }  
    }

    public class SHA256Hashing : HashingStrategy  
    {  
        public override void HashingInterface(string text)  
        {  
            SHA256 sha256 = new SHA256CryptoServiceProvider();

            Byte[] bytes;  
            bytes = ASCIIEncoding.Default.GetBytes(text);

            Byte[] encodedBytes;  
            encodedBytes = sha256.ComputeHash(bytes);

            string A = BitConverter.ToString(encodedBytes);  
        }  
    }

    public class SHA384Hashing : HashingStrategy  
    {  
        public override void HashingInterface(string text)  
        {  
            SHA384 sha384 = new SHA384CryptoServiceProvider();

            Byte[] bytes;  
            bytes = ASCIIEncoding.Default.GetBytes(text);

            Byte[] encodedBytes;  
            encodedBytes = sha384.ComputeHash(bytes);

            string A = BitConverter.ToString(encodedBytes);  
        }  
    }  
  
Client  
  
HashingContext context;  
  
context = new HashingContext(new SHA1Hashing());  
context.HashPassword("Chinna Sushma");  
  
context = new HashingContext(new SHA384Hashing());  
h.HashPassword("Chinna Lohetha");

* **Template Method:** Define an algorithm skeleton and delegate algorithm steps to subclasses so that they may be overridden
* **Visitor:** Add new operations to classes without modifying them

# What is the difference between Factory and Abstract Factory Patterns?

The common thing they have is that they belong to creational patterns. In short they hide the complexity of creating objects. The main difference between factory and Abstract factory is factory method uses inheritance to decide which object has to be instantiated while abstract factory uses delegation to decide instantiation of object. We can say Abstract factory uses factory method to complete the architecture. Abstract Factory is one level higher in abstraction over Factory.

# Project Management

Project Estimation

Estimating the effort, time, and resources needed to complete project activities is one of the most challenging tasks that project managers must face. This is because of the inherent uncertainty associated with many activities.I'll discuss three types of activities and what type of estimating approach should be used with each of them. Those are the [Stable Activities](http://www.projectmanagementguru.com/estimating.html#stable), the [Dependent Activities](http://www.projectmanagementguru.com/estimating.html#dependent), and the [Uncertain Activities.](http://www.projectmanagementguru.com/estimating.html#uncertain)

The techniques will include [Analogous](http://www.projectmanagementguru.com/estimating.html#analagous), [Parametric Modeling](http://www.projectmanagementguru.com/estimating.html#parametric), [3 Point Estimate](http://www.projectmanagementguru.com/estimating.html#3point), [Expert Judgment](http://www.projectmanagementguru.com/estimating.html#expert), [Published Data Estimates](http://www.projectmanagementguru.com/estimating.html#data), [Vendor Bid Analysis](http://www.projectmanagementguru.com/estimating.html#vendor), [Reserve Analysis](http://www.projectmanagementguru.com/estimating.html#reserve), [Bottom Up Analysis](http://www.projectmanagementguru.com/estimating.html#bottom), and [Simulation](http://www.projectmanagementguru.com/estimating.html#simulation). A discussion of estimating the cost of completing a project that is underway is addressed in the page on [Earned Value Analysis](http://www.projectmanagementguru.com/eva.html). Finally, I will discuss how to estimate a project when the key boundary condition is the [End Date](http://www.projectmanagementguru.com/estimating.html#end) or the [Total Cost](http://www.projectmanagementguru.com/estimating.html#cost) of the project and the effort is tailored to fit this constraint.

Stable Activities  
  
Stable Activities are those that are well understood and predictable. For activities in this category, the estimating is usually straightforward. I will typically use analogous, expert judgment, a parametric model, or published estimating data for these types of activities. Based upon the information available to the project team members, use the appropriate technique and set the estimate.  
  
Dependent Activities  
  
Dependent Activities are those activities where the time or effort is highly dependent upon some project attribute or characteristic that is not yet know or knowable at the time the original estimate is furnished. For instance, the amount of time needed to complete testing will depend upon whether the test is successful on the first try or whether a retest is required. For these types of activities, an assumption is made that will drive the estimated effort, time and resources. This assumption is a risk and should be tracked on the Risk Register. If the assumption is incorrect, the time or money required to do the activity may be very different from the estimate. If a conservative estimate is used, this is a positive risk. If an aggressive estimate is used, this is a negative risk.  
For these Dependent Activities, I often use the 3 Point Estimate, Expert Judgment, or Analogous Estimate. Also, if a project activity is outsourced and it is a Dependent Activity, I will consider what assumptions are used by the supplier and do a Vendor Bid Analysis. If there are a large number of Dependent Activities in the project, I will factor that into the project Reserve Analysis.  
  
Uncertain Activities  
  
Uncertain Activities are the most difficult to estimate. There is often very little data to support a precise estimate. In addition, there are many factors that could affect the estimate so I can't just make one assumption and track that in my risk register. An example of an Uncertain Activity is a requirements definition task on a [Complex](http://www.projectmanagementguru.com/complexity) project. There are numerous stakeholders who have different opinions of what is needed. Getting all of them to agree on the requirements will be an iterative process with the number of iterations being completely unpredictable. Yet if this task is not done well, there are likely to be major problems later in the project getting the stakeholders to agree that the project deliverables have been met. Uncertain Activities typically are listed in the Risk Register since the timing and cost are impossible to estimate accurately.  
  
For these Uncertain Activities, I often rely on the 3 Point Estimate to set the activity boundaries, although Published Data or Parametric Models can also be used to do this. Then I use Analogous or Expert Judgment to set the actual estimate. These activities must be considered in the Reserve Analysis. Often the estimate on these activities can be improved by decomposing the work of the activity and conducting a Bottom Up Analysis on that work. This will isolate the uncertain portions of the activity and allows for accurate estimates where possible.  
  
Analogous Estimating  
  
Analogous Estimating is one of the most common forms of estimating project activities. This technique uses the experience from previous projects and extrapolates that onto the current project. This technique is appropriate for those cases where the type of work is similar and the resources doing the work are the same between projects. Its advantage is that it is quick and, when the conditions are appropriate, it is usually fairly accurate. The disadvantage is that the organization must have similar projects for comparison.  
  
Parametric Model Estimating  
  
Parametric Model Estimating is a very accurate and easy estimating technique. A formula is developed for estimating the time or resources needed to perform a project activity. The formula is usually based upon a great deal of historical experience. A PMO will often develop the parametric model based upon having done lessons learned on many projects. A classic example from construction projects is the parametric model for estimating resources and time based upon the number of square feet of new construction. The advantage of parametric model estimating it is quick and accurate. The disadvantage is that models don't exist for activities until there is a large experience base for the activity.  
  
3 Point Estimating  
  
The 3 Point Estimating technique is used to understand the level of uncertainty embedded within an estimate. In this technique three estimates are generated for the project activity using three different sets of assumptions. The first estimate is a best case or optimistic estimate. The second estimate is a worst case or pessimistic estimate. The third estimate is between the other two and is the most likely estimate. The way those estimates are developed is by using one of the other techniques such as Analogous or Parametric Model. However, because of the high degree of uncertainty due to the risk assumptions, the three estimates are used to create a boundary on expectations for the activity. A variation on this technique, the PERT analysis, uses a weighted average of these estimates to create a PERT estimate. When using this approach, the most likely estimate is normally what is put in the project plan but the optimistic and pessimistic estimates are used during the reserve analysis. Also, an activity that has a great deal of difference between the optimistic and pessimistic estimates is an uncertain activity and should be tracked in the Risk Register. The advantage of this technique is that it provides boundaries on expectations. The disadvantages are that it takes more work - since three estimates must be created not one - and the most likely is still very much a guess - the actual could be significantly better or worse.  
  
Expert Judgment Estimating  
  
Expert Judgment estimating is easy to do - provided you have an expert on the project. This technique looks to the expert to create an estimate based upon their understanding of the project requirements. Many, if not most, project estimates are created in this fashion. The advantage of this is that it is quick and if the expert is knowledgeable, it is often the most accurate estimate for uncertain activities. The disadvantages are that you may not have an expert available and even if you do, the expert often can provide no solid rationale for their estimate beyond, "That's what I think it will take to do this."  
  
Published Data Estimating  
  
Published Data Estimating is an excellent technique for those activities for which there is published data. In this technique, the activity is compared to the activities for which data exists and the actual cost or durations of the closest comparable activity is selected from the data and used as the estimate. The advantage of this technique is that it is very accurate when the project conditions match the conditions under which the published data was generated. The disadvantages are that data does not exist for many activities and that the published data that does exist is based upon the characteristics of the organizations who compiled and published the data - which may not correspond with your organization's characteristics. (For instance you may have individuals on your project who are either much more or much less experienced than those who were in the projects comprising the data.)  
  
Vendor Bid Analysis  
  
The Vendor Bid Analysis is a technique used when working with suppliers on uncertain activities. The analysis considers the assumptions the vendor worked with and does a sensitivity assessment on those assumptions. In addition, for effort that the buying organization does not have experience with, they can contract with a consulting firm that has experience to do a "Should Cost" analysis. This "Should Cost" estimate is compared to the suppliers quote to identify any shortcomings. The advantages of this technique is that it exposes supplier risk that can be accounted for in the reserve analysis and it increases the confidence in the supplier's approach. The disadvantages are that this can take a fair amount of time and if a consultant is used to create a "Should Cost" it adds to the cost of the project.  
  
Reserve Analysis  
  
The Reserve Analysis is a fundamental technique for estimating. This technique considers the level of uncertainty and risk in the project and establishes a reserve pool of time, resources, or possibly performance that can be drawn upon to offset the unestimated issues that arise. With respect to schedule reserve, I will allocate the path float that I have on non-critical path tasks to provide a buffer around those tasks with the most uncertain estimates. For critical path tasks, I will either use a conservative estimate or include a "reserve" task that is used to resolve any problems that arise. (An example of a reserve task is "bug-fixing" on a software development project. I have been asked, "Why don't you just write software that doesn't have bugs. Then you won't need that task." Of course I don't intend to write software with bugs. However, experience has shown that the software is likely to have some, so I add a "bug-fixing" task as a reserve task to address the bugs. The specific bugs are not known when the task is created, but based upon the level of uncertainty in the software development, an amount of time and resources are set aside to resolve software development problems.) With respect to resource reserves, I will set the level of reserve based upon the uncertainty in the project and the risks. It is common to have resource reserves of 10% to 20% over the estimated cost of a project for those projects with high uncertainty. Although projects with low uncertainty might have 0% to 5% reserve. Again I allocate that reserve to the areas of highest uncertainty. I also will allocate a portion of the resource reserve to each phase in a phased project. This is to better match the actual the planned timing for the use of the reserve with when it is likely to be used. This avoids unnecessary variance reports. The use of performance reserves is when task deliverables are expanded to address unknown issues. For instance, we would typically design with a 50% design margin on our mechanical device designs so that if there was a slight error in calculations or material properties, the design would still be robust enough to perform as needed. At the end of the day, the reserve analysis is a guess on the part of the project manager and stakeholders. If your organization has policies regarding reserves, use them. If not, use your best judgment.  
  
Bottom Up Analysis  
  
A Bottom Up analysis is a technique to improve the accuracy of the overall project estimate. This technique requires the project team to decompose the work into very small work packages. Generally, the smaller the project activity, the easier it is to estimate because the work scope is very small. All of these estimates of small activities are added up into subgroups and finally into the project total. The advantage of this technique is that the estimate is usually more accurate since the work is better understood. The disadvantages of this technique are that it is very time consuming, and it may be impossible to decompose activities that cannot be easily defined.  
  
Project Simulation  
  
A Project Simulation is a way of combining the uncertainty in 3 Point Estimates and Reserve Analysis to understand the likely project outcomes. This requires that the project be entered into a project management software application, being careful to identify all relationships between activities. For those activities that have uncertainty, the degree of uncertainty must be modeled and entered into the project management software also. The method for doing this will vary based upon the software being used. A simulation add-on is then used to run the software through a Monte Carlo routine. The result will be a distribution of project time lines and costs. Based upon the organization's risk sensitivity, the overall project time line and budget can be set. The advantage of this approach is that it provides a global perspective on overall time line and cost uncertainty. The disadvantages are that it can take months to do this on a large project and the resulting estimates are only as good as the assumptions that are allowed by the software. In my experience, the software is seldom able to truly model all of the uncertainty and relationships.  
  
Estimating Based upon Project End Date  
  
In some cases, the project end date is set even before the scope and deliverables are defined. (I am reminded of Y2K projects.) In those cases, a high-level time line is created starting from the end date and going backward to the present time. Given the amount of time allocated for the major activities, the project team considers the needed deliverables and available resources during the time period. Essentially, the schedule side of the triangle is fixed and the scope and resource sides are varied so as to create a viable project. Often this will require an iterative estimating approach. Once the high level plan is established, estimates for the activities are developed and then iterations are done varying resources and scope until a viable estimate can be created. The Risk Register will be dominated by schedule risk items. Sometimes, an estimate cannot be created. In those cases, the project should not even be initiated, since it is doomed.  
  
Estimating Based Upon Project Total Cost  
  
In some cases, the project total cost is set even before the scope and schedule are defined. (I am reminded of construction projects funded by bank loans.) In those cases, a high-level allocation of the budget is created between the likely project deliverables. Each major activity is then estimated and if the estimate is greater than the allocated cost, the timing of resources or scope and deliverables are varied until the project is able to meet the budget goals. This is often an iterative process that may take much iteration until it completes.

# Reference

<http://www.mindtools.com/pages/article/newPPM_01.htm>