jquery-

jQuery is a JavaScript Library.
jQuery greatly simplifies JavaScript programming.
jQuery is easy to learn.

What is jQuery?

jQuery is a lightweight, "write less, do more", JavaScript library.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

HTML/DOM manipulation
CSS manipulation
HTML event methods
Effects and animations
AJAX
Utilities

Why jQuery?

There are lots of other JavaScript libraries out there, but jQuery is probably the most popular, and also the most extendable.

Many of the biggest companies on the Web use jQuery, such as:

Google
Microsoft
IBM
Netflix

****

CDN-

A content delivery network (CDN) refers to a geographically distributed group of servers which work together to provide fast delivery of Internet content.

A CDN allows for the quick transfer of assets needed for loading Internet content including HTML pages, javascript files, stylesheets, images, and videos. The popularity of CDN services continues to grow, and today the majority of web traffic is served through CDNs, including traffic from major sites like Facebook, Netflix, and Amazon.

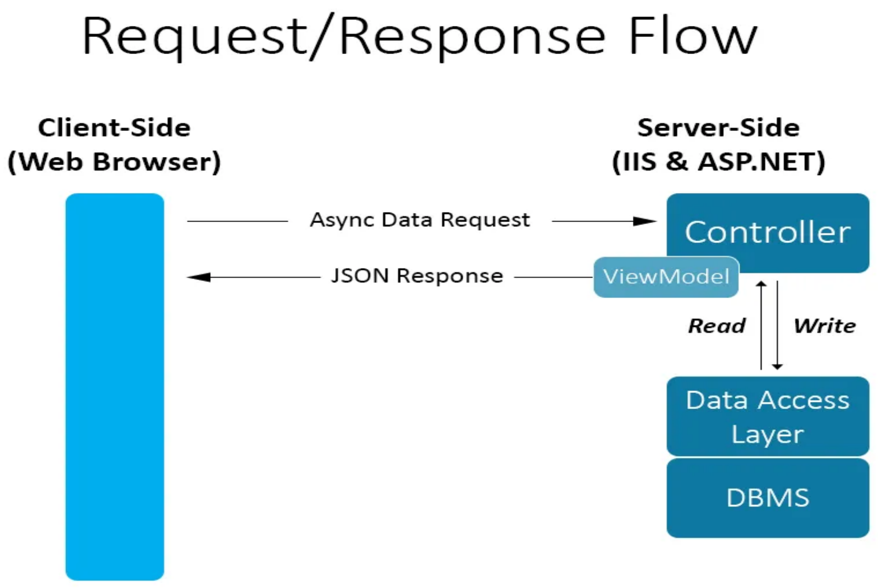
A properly configured CDN may also help protect websites against some common malicious attacks, such as Distributed Denial of Service (DDOS) attacks.

****

The HTTP Request-Response Data Flow in Native Web Applications -

As most of us already know, the Hypertext Transfer Protocol (HTTP) is designed to enable communications between clients and servers. HTTP works as a request-response protocol between a client and server: in a standard WWW-based scenario the client is usually a web browser, while the server role is played by a Web Application hosted on a remote computer called the Web Server.

The browser submits a HTTP request to the server; that request is processed by the server, which returns a HTTP response to the client. The response contains status information about the request and may also contain the requested content.



What is the difference between session and cookies?

Cookies and Sessions are used to store information. Cookies are only stored on the client-side machine, while sessions get stored on the client as well as a server.

Session

A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit.

A session ends when the user closes the browser or after leaving the site, the server will terminate the session after a predetermined period of time, commonly 30 minutes duration.

Cookies

Cookies are text files stored on the client computer and they are kept of use tracking purpose. Server script sends a set of cookies to the browser. For example name, age, or identification number etc. The browser stores this information on a local machine for future use.

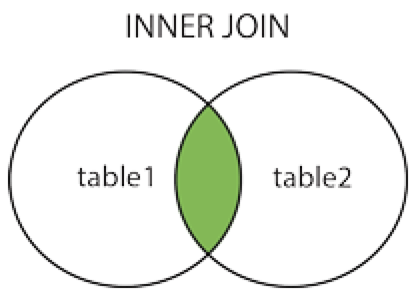
When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user.

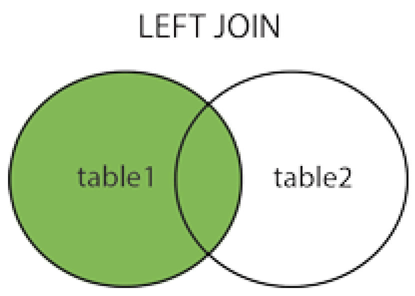
stored procedure

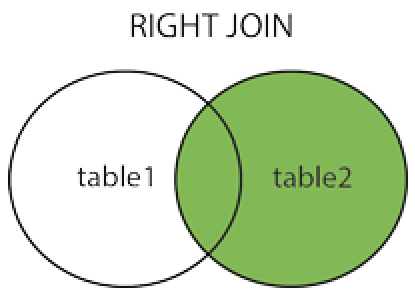
A stored procedure is a set of Structured Query Language (SQL) statements with an assigned name, which are stored in a relational database management system (RDBMS) as a group, so it can be reused and shared by multiple programs.

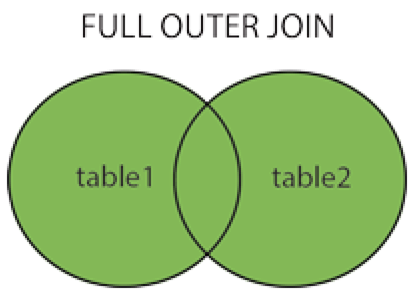
Stored procedures can access or modify data in a database, but it is not tied to a specific database or object, which offers a number of advantages.

SQL JOIN









Here are the different types of the JOINs in SQL:

- (INNER) JOIN: Returns records that have matching values in both tables

```
        Select column_1, column_2, column_3 FROM table_1 INNER JOIN table_2 ON
table_1.column = table_2.column;
```

- LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table

```
        Select column_1, column_2, column(s) FROM table_1 LEFT JOIN table_2 ON
table_1.column_name = table_2.column_name;
```

- RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table

  Select column_1, column_2, column(s) FROM table_1 RIGHT JOIN table_2 ON table_1.column_name = table_2.column_name;

- FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table

Select column_1, column_2, column(s) FROM table_1 FULL JOIN table_2 ON table_1.column_name = table_2.column_name;

CROSS JOIN

It is also known as **CARTESIAN JOIN**, which returns the Cartesian product of two or more joined tables. The **CROSS JOIN** produces a table that merges each row from the first table with each second table row. It is not required to include any condition in CROSS JOIN.

1. Select * from table_1 cross join table_2;

Log4net

Log4net is a logging framework for the .NET platform. It's definitely not the only one, but it's one of the most popular frameworks out there.

A logging framework is a tool that can dramatically reduce the burden of dealing with logs.

abstract class

An abstract class is a special type of class that cannot be instantiated. An abstract class is designed to be inherited by subclasses that either implement or override its methods. In other words, abstract classes are either partially implemented or not implemented at all. You can have functionality in your abstract class—the methods in an abstract class can be both abstract and concrete. An abstract class can have constructors—this is one major difference between an abstract class and an interface. You can take advantage of abstract classes to design components and specify some level of common functionality that must be implemented by derived classes.

ERROR HANDLING

1.Proactive approach

Errors

Programming errors where there is no way to recover/continue gracefully and usually need a programmer to step into and change the code to make the fix. Errors can sometimes be turned into exceptions so that they can be handled within the code.

Error handling best practices

Errors can usually be avoided with simple checks and if simple checks won't suffice errors can also turn into exceptions, so that the application can handle the situation gracefully.

4 main ones I know: try/catch, explicit returns, either, and supervising crashes.

Object Relational Mapping (ORM)

When we work with an object-oriented system, there is a mismatch between the object model and the relational database. RDBMSs represent data in a tabular format whereas object-oriented languages, such as Java or C# represent it as an interconnected graph of objects.

ORM stands for **O**bject-**R**elational **M**apping (ORM) is a programming technique for converting data between relational databases and object oriented programming languages such as Java, C#, etc.

Exception Handling

Exceptions provide a way to transfer control from one part of a program to another. C# exception handling is built upon four keywords: **try**, **catch**, **finally**, and **throw**.

- **try** − A try block identifies a block of code for which particular exceptions is activated. It is followed by one or more catch blocks.
- **catch** − A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The catch keyword indicates the catching of an exception.
- **finally** − The finally block is used to execute a given set of statements, whether an exception is thrown or not thrown. For example, if you open a file, it must be closed whether an exception is raised or not.

- **throw** − A program throws an exception when a problem shows up. This is done using a throw keyword.

System.IO.IOException

Handles I/O errors.

System.IndexOutOfRangeException

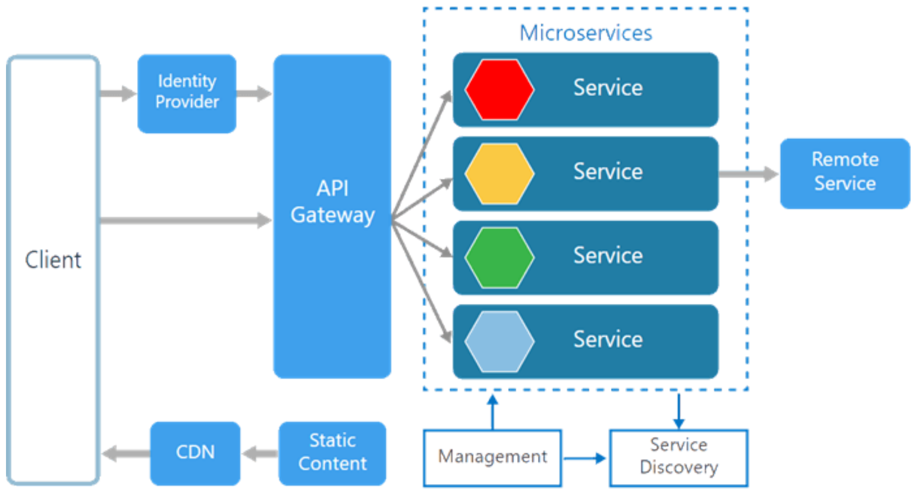
Handles errors generated when a method refers to an array index out of range.

System.DivideByZeroException

Handles errors generated from dividing a dividend with zero.

Microservices

The term microservices portrays a software development style that has grown from contemporary trends to set up practices that are meant to increase the speed and efficiency of developing and managing software solutions at scale. Microservices is more about applying a certain number of principles and architectural patterns as architecture. Each microservice lives independently, but on the other hand, also all rely on each other. All microservices in a project get deployed in production at their own pace, on-premise on the cloud, independently, living side by side..



Management. Maintains the nodes for the service.

Identity Provider. Manages the identity information and provides authentication services within a distributed network.

Service Discovery. Keeps track of services and service addresses and endpoints.

API Gateway. Serves as client's entry point. Single point of contact from the client which in turn returns responses from underlying microservices and sometimes an aggregated response from multiple underlying microservices.

CDN. A content delivery network to serve static resources for e.g. pages and web content in a distributed network

Static Content The static resources like pages and web content

microservice program

Design Pattern

Design patterns represent the best practices used by experienced object-oriented software developers. Design patterns are solutions to general problems that software developers faced during software development. These solutions were obtained by trial and error by numerous software developers over quite a substantial period of time.

Creational Patterns

These design patterns provide a way to create objects while hiding the creation logic, rather than instantiating objects directly using new operator. This gives program more flexibility in deciding which objects need to be created for a given use case.

Structural Patterns

These design patterns concern class and object composition. Concept of inheritance is used to compose interfaces and define ways to compose objects to obtain new functionalities.

Behavioral Patterns

These design patterns are specifically concerned with communication between objects.

J2EE Patterns

These design patterns are specifically concerned with the presentation tier. These patterns are identified by Sun Java Center.

Dependency Injection

Dependency Injection (DI) is a software design pattern that allows 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.

Triggers

Triggers are stored programs, which are automatically executed or fired when some events occur. Triggers are, in fact, written to be executed in response to any of the following events −

- A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE)
- A **database definition (DDL)** statement (CREATE, ALTER, or DROP).
- A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

Triggers can be defined on the table, view, schema, or database with which the event is associated.

Benefits of Triggers

Triggers can be written for the following purposes −

- Generating some derived column values automatically
- Enforcing referential integrity
- Event logging and storing information on table access
- Auditing
- Synchronous replication of tables
- Imposing security authorizations
- Preventing invalid transactions

Delegates

C# delegates are similar to pointers to functions, in C or C++. A **delegate** is a reference type variable that holds the reference to a method. The reference can be changed at runtime.

Delegates are especially used for implementing events and the call-back methods. All delegates are implicitly derived from the **System.Delegate** class.

Declaring Delegates

public delegate int MyDelegate (string s);

Bundling and Minification

Bundling and minification techniques were introduced in MVC 4 to improve request load time. Bundling allows us to load the bunch of static files from the server in a single HTTP request.

Minification

Minification technique optimizes script or CSS file size by removing unnecessary white space and comments and shortening variable names to one character.

Bundle Types

MVC 5 includes following bundle classes in System.web.Optimization namespace:

ScriptBundle: ScriptBundle is responsible for JavaScript minification of single or multiple script files.

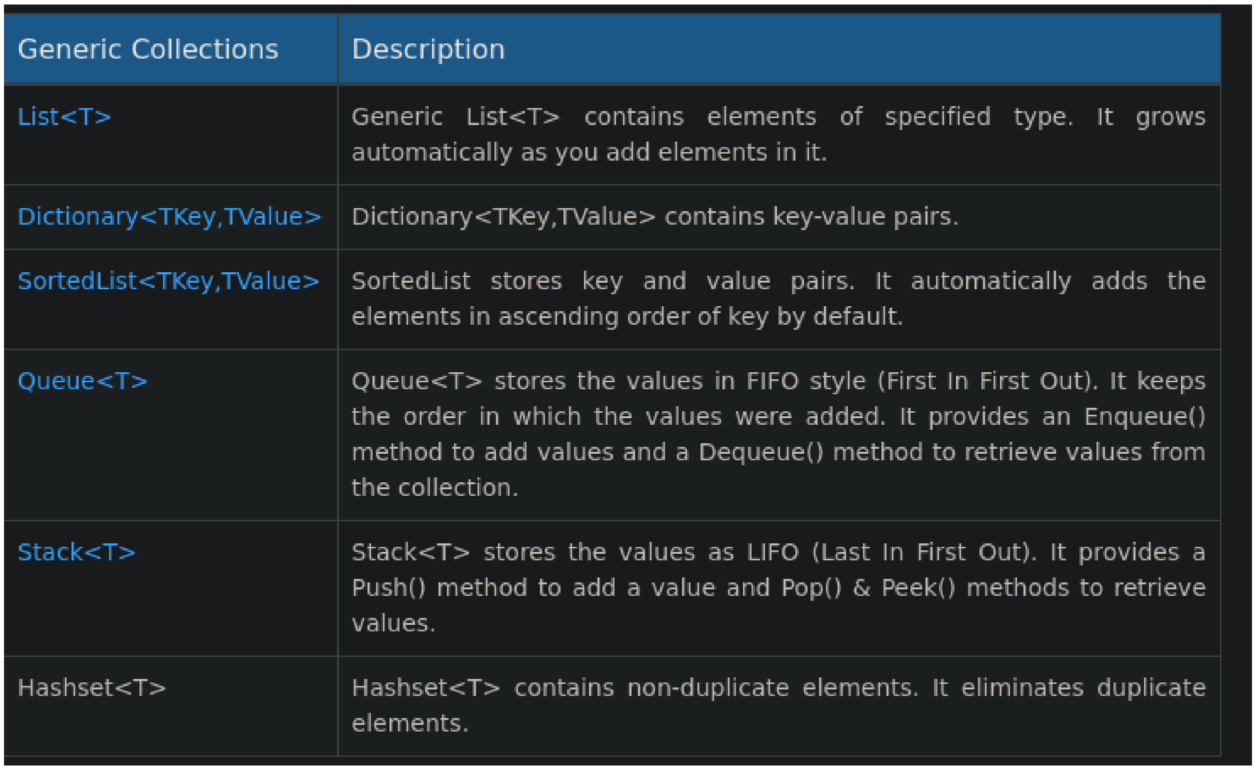
StyleBundle: StyleBundle is responsible for CSS minification of single or multiple style sheet files.

DynamicFolderBundle: Represents a Bundle object that ASP.NET creates from a folder that contains files of the same type.

Generic Collections In C#

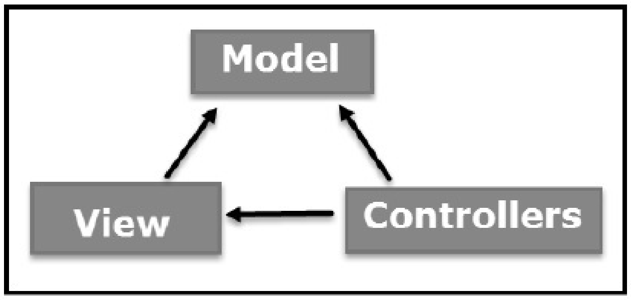
C# includes specialized classes that store series of values or objects are called collections.

A generic collection is strongly typed (you can store one type of objects into it) so that we can eliminate runtime type mismatches, it improves the performance by avoiding boxing and unboxing.



Generic Collections	Description
List<T>	Generic List<T> contains elements of specified type. It grows automatically as you add elements in it.
Dictionary<TKey,TValue>	Dictionary<TKey,TValue> contains key-value pairs.
SortedList<TKey,TValue>	SortedList stores key and value pairs. It automatically adds the elements in ascending order of key by default.
Queue<T>	Queue<T> stores the values in FIFO style (First In First Out). It keeps the order in which the values were added. It provides an Enqueue() method to add values and a Dequeue() method to retrieve values from the collection.
Stack<T>	Stack<T> stores the values as LIFO (Last In First Out). It provides a Push() method to add a value and Pop() & Peek() methods to retrieve values.
Hashset<T>	Hashset<T> contains non-duplicate elements. It eliminates duplicate elements.

MVC Framework



Model

The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. For example, a Customer object will retrieve the customer information from the database, manipulate it and update it data back to the database or use it to render data.

View

The View component is used for all the UI logic of the application. For example, the Customer view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with.

Controller

Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data.

ASP.NET MVC

ASP.NET supports three major development models: Web Pages, Web Forms and MVC (Model View Controller). ASP.NET MVC framework is a lightweight, highly testable presentation framework that is integrated with the existing ASP.NET features, such as master pages, authentication, etc. Within .NET, this framework is defined in the System.Web.Mvc assembly. The latest version of the MVC Framework is 5.0. We use Visual Studio to create ASP.NET MVC applications which can be added as a template in Visual Studio.

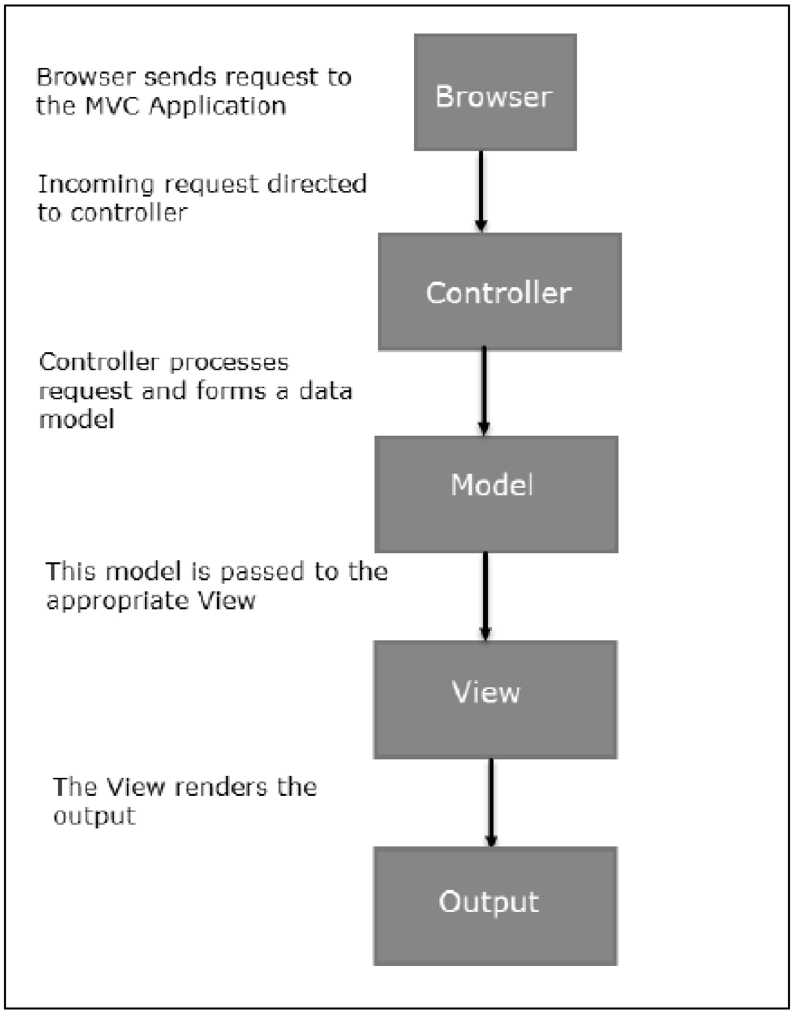
ASP.NET MVC Features

ASP.NET MVC provides the following features −

- Ideal for developing complex but lightweight applications.
- Provides an extensible and pluggable framework, which can be easily replaced and customized. For example, if you do not wish to use the in-built Razor or ASPX View Engine, then you can use any other third-party view engines or even customize the existing ones.

- Utilizes the component-based design of the application by logically dividing it into Model, View, and Controller components. This enables the developers to manage the complexity of large-scale projects and work on individual components.
- MVC structure enhances the test-driven development and testability of the application, since all the components can be designed interface-based and tested using mock objects. Hence, ASP.NET MVC Framework is ideal for projects with large team of web developers.
- Supports all the existing vast ASP.NET functionalities, such as Authorization and Authentication, Master Pages, Data Binding, User Controls, Memberships, ASP.NET Routing, etc.
- Does not use the concept of View State (which is present in ASP.NET). This helps in building applications, which are lightweight and gives full control to the developers.

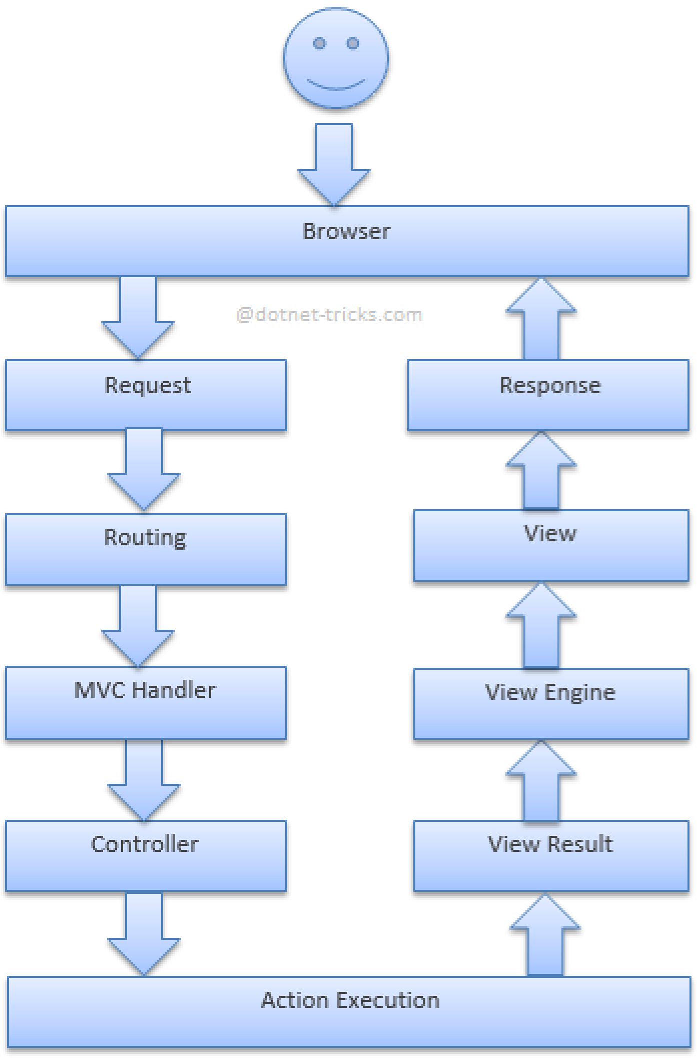
Thus, you can consider MVC Framework as a major framework built on top of ASP.NET providing a large set of added functionality focusing on component-based development and testing.


Browser sends request to the MVC Application
Browser
Incoming request directed to controller
Controller
Controller processes request and forms a data model
Model
This model is passed to the appropriate View
View
The View renders the output
Output

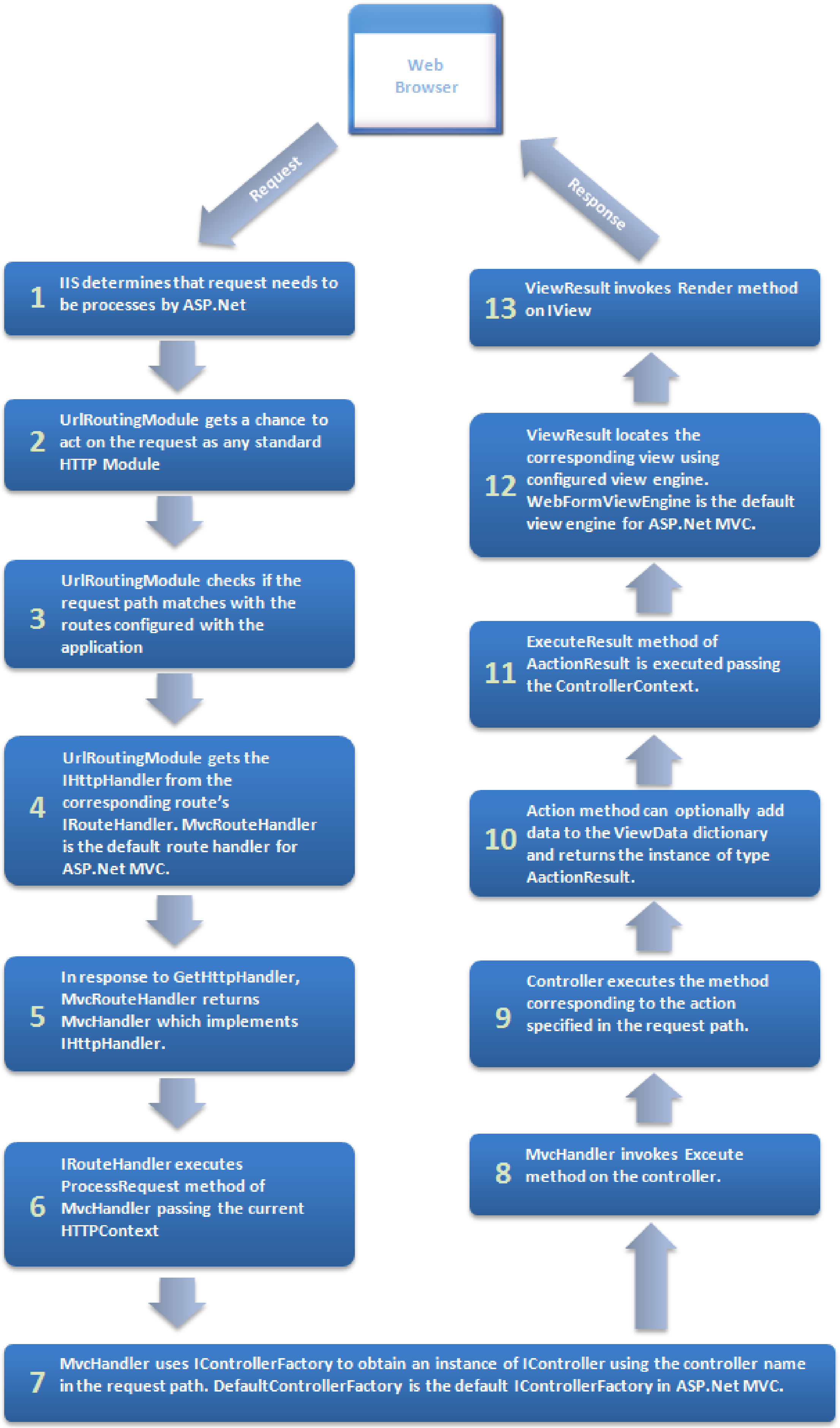
1. MVC stands for Model, View and Controller.
2. Model represents the data
3. View is the User Interface.
4. Controller is the request handler.

Model: Model represents the shape of the data. A class in C# is used to describe a model. Model objects store data retrieved from the database.

ASP.NET MVC Page Life Cycle:



ASP.NET MVC Request Life Cycle


Web Browser
Request
Response
1 IIS determines that request needs to be processes by ASP.Net
2 UrlRoutingModule gets a chance to act on the request as any standard HTTP Module
3 UrlRoutingModule checks if the request path matches with the routes configured with the application
4 UrlRoutingModule gets the IHttpHandler from the corresponding route's IRouteHandler. MvcRouteHandler is the default route handler for ASP.Net MVC.
5 In response to GetHttpHandler, MvcRouteHandler returns MvcHandler which implements IHttpHandler.
6 IRouteHandler executes ProcessRequest method of MvcHandler passing the current HTTPContext
7 MvcHandler uses IControllerFactory to obtain an instance of IController using the controller name in the request path. DefaultControllerFactory is the default IControllerFactory in ASP.Net MVC.
8 MvcHandler invokes Exceute method on the controller.
9 Controller executes the method corresponding to the action specified in the request path.
10 Action method can optionally add data to the ViewData dictionary and returns the instance of type AactionResult.
11 ExecuteResult method of AactionResult is executed passing the ControllerContext.
12 ViewResult locates the corresponding view using configured view engine. WebFormViewEngine is the default view engine for ASP.Net MVC.
13 ViewResult invokes Render method on IView

Stored Procedure

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

Stored Procedure Syntax

```
CREATE PROCEDURE procedure_name

AS

sql_statement

GO;
```

Execute a Stored Procedure

```
EXEC procedure_name;
```

SOLID

SOLID is an acronym for the first five object-oriented design (OOD)

These principles establish practices that lend to developing software with considerations for maintaining and extending as the project grows. Adopting these practices can also contribute to avoiding code smells, refactoring code, and Agile or Adaptive software development.

SOLID stands for:

-
- **S** - Single-responsiblity Principle
- A class should have one and only one reason to change, meaning that a class should have only one job.
-
- **O** - Open-closed Principle
- Objects or entities should be open for extension but closed for modification.
-
-

-
- **L** - Liskov Substitution Principle
- Let q(x) be a property provable about objects of x of type T. Then q(y) should be provable for objects y of type S where S is a subtype of T.
-
- **I** - Interface Segregation Principle
- A client should never be forced to implement an interface that it doesn't use, or clients shouldn't be forced to depend on methods they do not use.
-
- **D** - Dependency Inversion Principle
- Entities must depend on abstractions, not on concretions. It states that the high-level module must not depend on the low-level module, but they should depend on abstractions.

Unit Testing

UNIT TESTING is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object

Migration

Entity Framework introduced a migration tool that automatically updates the database schema when your model changes without losing any existing data or other database objects. It uses a new database initializer called ***MigrateDatabaseToLatestVersion***.

There are two kinds of Migration:

1. Automated Migration
2. Code-based Migration

Code First to an Existing Database

(entity framework- to connect to the database)

https://www.tektutorialshub.com/entity-framework/ef-database-connection/

in-memory database

An in-memory database is a type of purpose-built database that relies primarily on memory for data storage, in contrast to databases that store data on disk or SSDs. In-memory databases are designed to attain minimal response time by eliminating the need to access disks. Because all data is stored and managed exclusively in main memory, it is at risk of being lost upon a process or server failure. In-memory databases can persist data on disks by storing each operation in a log or by taking snapshots.

In-memory databases are ideal for applications that require microsecond response times and can have large spikes in traffic coming at any time such as gaming leaderboards, session stores, and real-time analytics.

sealed (C# Reference)

When applied to a class, the sealed modifier prevents other classes from inheriting from it. In the following example, class B inherits from class A, but no class can inherit from class B.

```
class A {}
sealed class B : A {}
```

Access Modifiers in C#

Access Modifiers are keywords that define the accessibility of a member, class or datatype in a program. These are mainly used to restrict unwanted data manipulation by external programs or classes. There are **4** access modifiers (public, protected, internal, private) which defines the **6 accessibility levels** as follows:

public

protected

internal

protected internal

private

private protected

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

```
body {

  background-color: lightblue;

}



h1 {

  color: navy;

  margin-left: 20px;

}
```

Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

The internal style is defined inside the <style> element, inside the head section.

Inline CSS

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

```
<h1 style="color:blue;text-align:center;">This is a heading</h1>

<p style="color:red;">This is a paragraph.</p>
```

JavaScript Variables

```
<script>

var price1 = 5;

var price2 = 6;

var total = price1 + price2;

document.getElementById("demo").innerHTML =

"The total is: " + total;

</script>
```

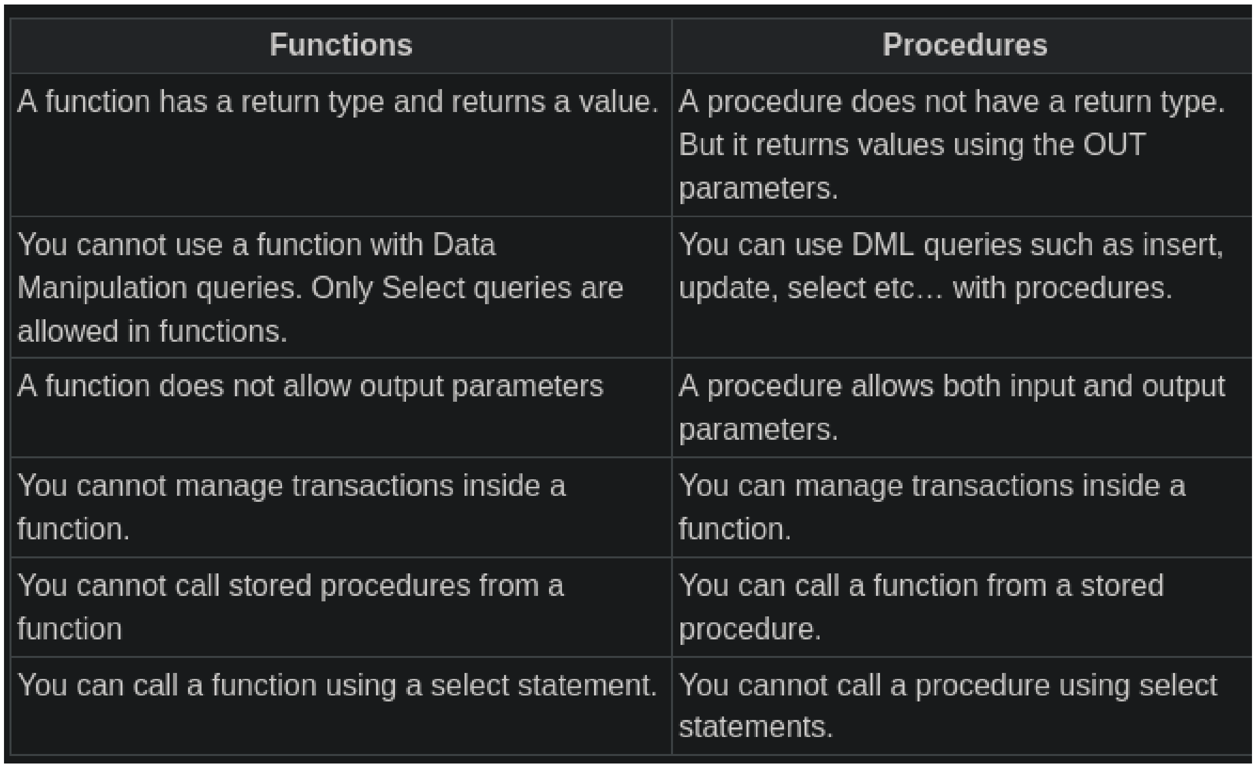
DDL is short name of **Data Definition Language,** which deals with database schemas and descriptions, of how the data should reside in the database.

- CREATE - to create a database and its objects like (table, index, views, store procedure, function, and triggers)
- ALTER - alters the structure of the existing database
- DROP - delete objects from the database
- TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed
- COMMENT - add comments to the data dictionary
- RENAME - rename an object

DML is short name of **Data Manipulation Language** which deals with data manipulation and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE, etc., and it is used to store, modify, retrieve, delete and update data in a database.

- SELECT - retrieve data from a database
- INSERT - insert data into a table
- UPDATE - updates existing data within a table
- DELETE - Delete all records from a database table
- MERGE - UPSERT operation (insert or update)
- CALL - call a PL/SQL or Java subprogram
- EXPLAIN PLAN - interpretation of the data access path
- LOCK TABLE - concurrency Control

Stored procedures and functions



Functions	Procedures
A function has a return type and returns a value.	A procedure does not have a return type. But it returns values using the OUT parameters.
You cannot use a function with Data Manipulation queries. Only Select queries are allowed in functions.	You can use DML queries such as insert, update, select etc… with procedures.
A function does not allow output parameters	A procedure allows both input and output parameters.
You cannot manage transactions inside a function.	You can manage transactions inside a function.
You cannot call stored procedures from a function	You can call a function from a stored procedure.
You can call a function using a select statement.	You cannot call a procedure using select statements.

Aggregate functions in SQL

In database management an aggregate function is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

Various Aggregate Functions

1) Count()

2) Sum()

3) Avg()

4) Min()

5) Max()

Debugging stored procedures

Debugging is one of the most important but painful parts of any software process. To find some errors you have to run the code step by step to see which section of the code is responsible for the error. This is called runtime debugging.

Luckily, SQL Server Management Studio (SSMS) comes with automated debugging capabilities to help developers debug their scripts.

Also uses breakpoints.

SQL Server TRY CATCH overview

The TRY CATCH construct allows you to gracefully handle exceptions in SQL Server. To use the TRY CATCH construct, you first place a group of Transact-SQL statements that could cause an exception in a BEGIN TRY...END TRY block as follows:

```
BEGIN TRY

   -- statements that may cause exceptions

END TRY

BEGIN CATCH

   -- statements that handle exception

END CATCH
```

The CATCH block functions

Inside the CATCH block, you can use the following functions to get the detailed information on the error that occurred:

- ERROR_LINE() returns the line number on which the exception occurred.
- ERROR_MESSAGE() returns the complete text of the generated error message.
- ERROR_PROCEDURE() returns the name of the stored procedure or trigger where the error occurred.
- ERROR_NUMBER() returns the number of the error that occurred.
- ERROR_SEVERITY() returns the severity level of the error that occurred.
- ERROR_STATE() returns the state number of the error that occurred.

Note that you only use these functions in the CATCH block. If you use them outside of the CATCH block, all of these functions will return NULL.

Example-

```
CREATE PROC usp_divide(
    @a decimal,
    @b decimal,
    @c decimal output
) AS
BEGIN
    BEGIN TRY
        SET @c = @a / @b;
    END TRY
    BEGIN CATCH
        SELECT
            ERROR_NUMBER() AS ErrorNumber
            ,ERROR_SEVERITY() AS ErrorSeverity
            ,ERROR_STATE() AS ErrorState
            ,ERROR_PROCEDURE() AS ErrorProcedure
            ,ERROR_LINE() AS ErrorLine
            ,ERROR_MESSAGE() AS ErrorMessage;
    END CATCH
END;
GO
```

Modifying an existing SQL Server stored procedure

To change the stored procedure and save the updated code you would use the ALTER PROCEDURE command as follows.

SQL Join vs Subquery

What are Joins?

A join is a query that combines records from two or more tables. A join will be performed whenever multiple tables appear in the FROM clause of the query. The select list of the query can select any columns from any of these tables. If join condition is omitted or invalid then a Cartesian product is formed. If any two of these tables have a column name in common, then must qualify these columns throughout the query with table or table alias names to avoid ambiguity. Most join queries contain at least one join condition, either in the FROM clause or in the WHERE clause.

what is Subquery?

A Subquery or Inner query or Nested query is a query within SQL query and embedded within the WHERE clause. A Subquery is a SELECT statement that is embedded in a clause of another SQL statement. They can be very useful to select rows from a table with a condition that depends on the data in the same or another table. A Subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved. The subquery can be placed in the following SQL clauses they are WHERE clause, HAVING clause, FROM clause.

SQL Cross Join

The SQL CROSS JOIN produces a result set which is the number of rows in the first table multiplied by the number of rows in the second table if no WHERE clause is used along with CROSS JOIN.This kind of result is called as Cartesian Product.

If WHERE clause is used with CROSS JOIN, it functions like an INNER JOIN.

Syntax:

```
SELECT *

FROM table1

CROSS JOIN table2;
```

Second max sal

```
SELECT name, MAX(salary) AS salary

  FROM employee

 WHERE salary < (SELECT MAX(salary)

                 FROM employee);
```

ADO.NET

ADO.NET provides a bridge between the front end controls and the back end database. The ADO.NET objects encapsulate all the data access operations and the controls interact with these objects to display data, thus hiding the details of movement of data.

Web API

In computer programming, an application programming interface (API) is a set of subroutine definitions, protocols, and tools for building software and applications.

To put it in simple terms, API is some kind of interface which has a set of functions that allow programmers to access specific features or data of an application, operating system or other services.

ASP.NET - Managing State

ASP.NET manages four types of states:

- View State
- Control State
- Session State
- Application State

Code First v/s Database First

Code First Approach

In code first approach we will first create entity classes with properties defined in it. Entity framework will create the database and tables based on the entity classes defined. So database is generated from the code. When the dot net code is run database will get created.

Advantages

1. You can create the database and tables from your business objects.
2. You can specify which related collections are to be eager loaded, or not be serialized at all.

3. Database version control.
4. Good for small applications.

Database First Approach

In this approach Database and tables are created first. Then you create entity Data Model using the created database.

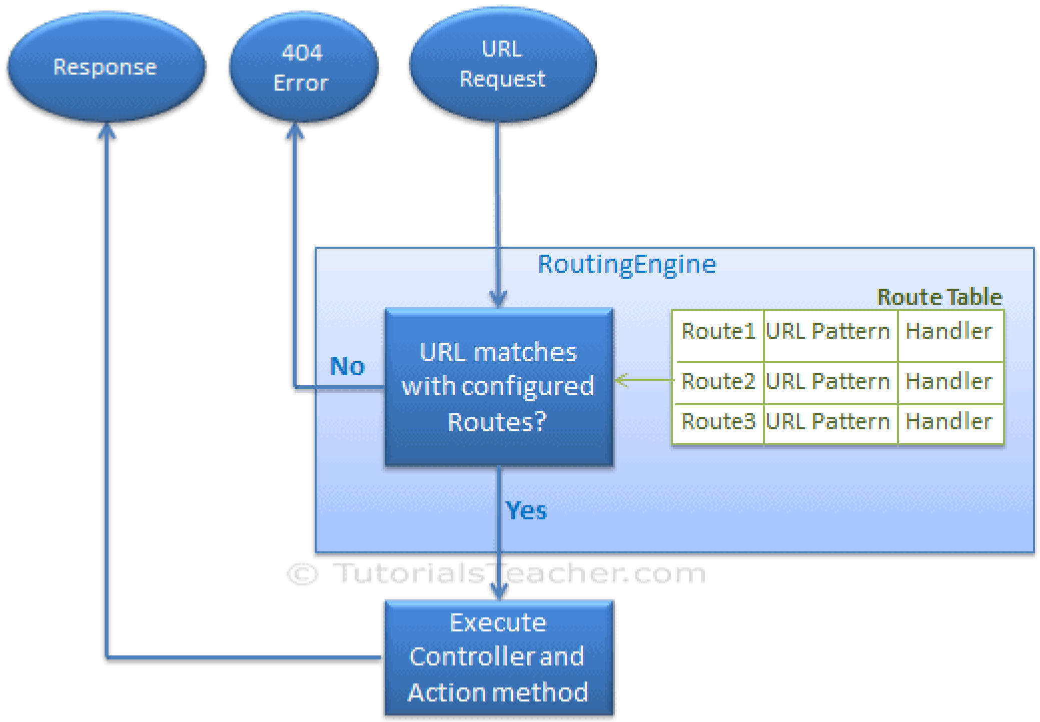
Advantages

1. Simple to create the data model

2. Graphical user interface.
3. Mapping and creation of keys and relationships are easy as you need not have to write any code .
4. Preferred for data intense and large applications

Routing in MVC

In the ASP.NET Web Forms application, every URL must match with a specific .aspx file. For example, a URL http://domain/studentsinfo.aspx must match with the file studentsinfo.aspx that contains code and markup for rendering a response to the browser.

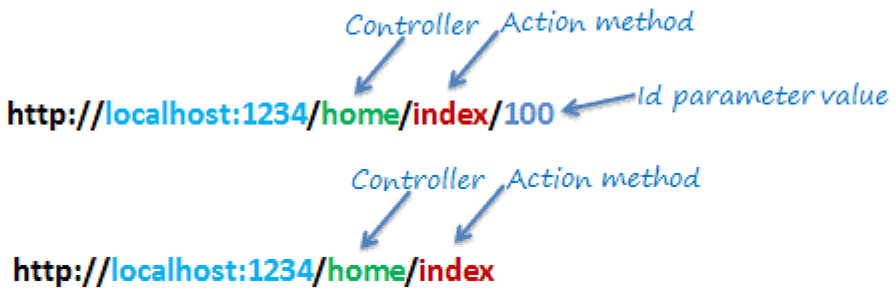

Response
404 Error
URL Request
RoutingEngine
Route Table
Route1 URL Pattern Handler
Route2 URL Pattern Handler
Route3 URL Pattern Handler
No
URL matches with configured Routes?
Yes
© TutorialsTeacher.com
Execute Controller and Action method

Configure a Route

Every MVC application must configure (register) at least one route configured by the MVC framework by default. You can register a route in RouteConfig class, which is in RouteConfig.cs under App_Start folder. The following figure illustrates how to configure a route in the RouteConfig class .

URL Pattern

The URL pattern is considered only after the domain name part in the URL. For example, the URL pattern *"{controller}/{action}/{id}"* would look like localhost:1234/{controller}/{action}/{id}. Anything after "localhost:1234/" would be considered as a controller name.


Controller
Action method
http://localhost:1234/home/index/100
Id parameter value
Controller
Action method
http://localhost:1234/home/index

Why entity framework use over ado.net.

I suggest you to go with the Entity Framework(EF) because EF is an object-relational mapper that allows the developers to interact with the relational data using the domain-space objects.It works on the top on the ADO.NET by handling the ADO.NET provider.

The ADO.NET provides the dataset, datatables, command, and connection objects.It allows you to work on the higher level objects such as managers,workers,suppliers etc..

The ADO.NET provides the set of software components that the developer can use access data and various data services in the database.

Approaches of Entity Framework

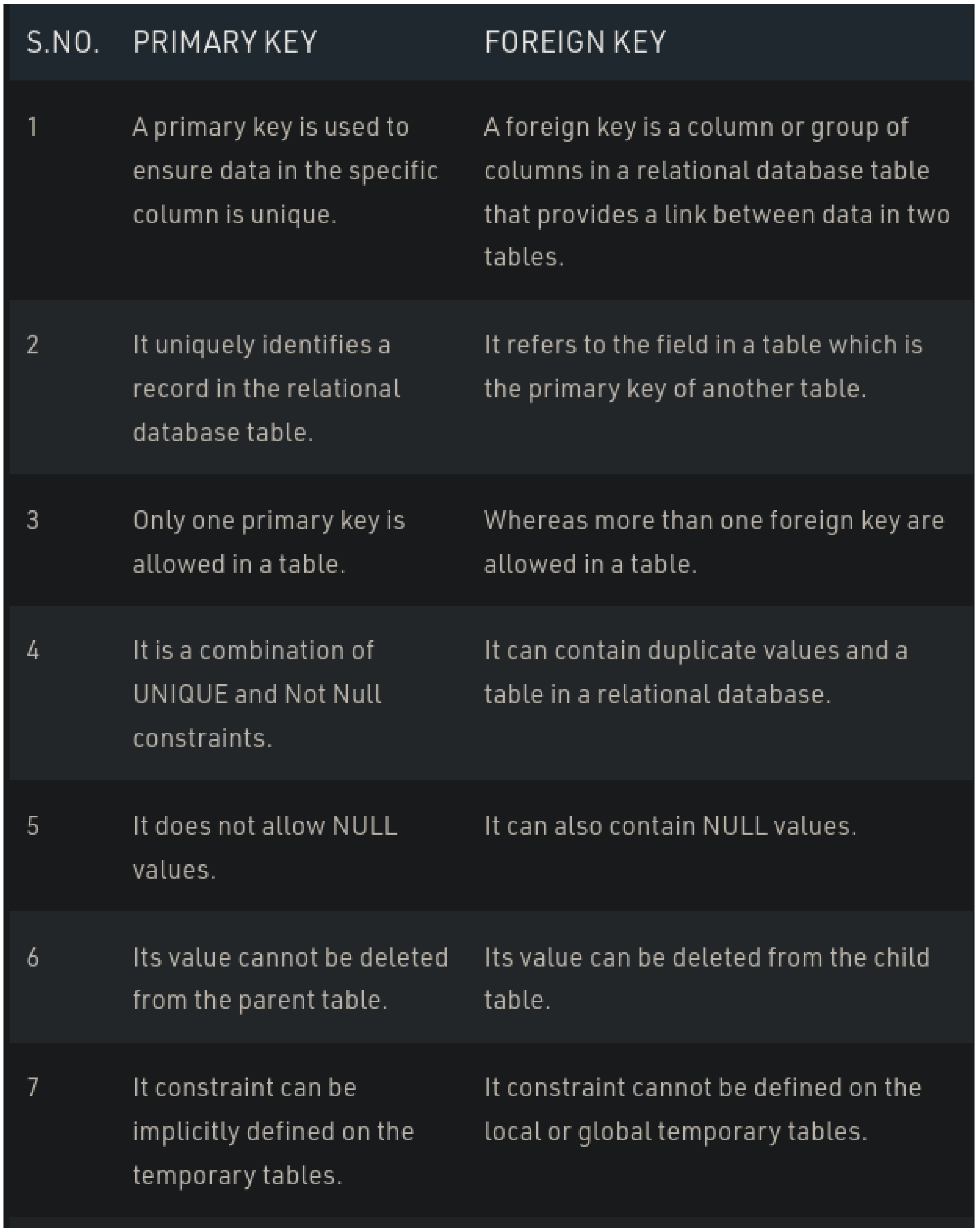
1.model view
2.code first
3.database first

Difference between Primary Key and Foreign Key-
Primary Key:

A primary key is used to ensure data in the specific column is unique. It is a column cannot have NULL values. It is either an existing table column or a column that is specifically generated by the database according to a defined sequence.

Foreign Key:

A foreign key is a column or group of columns in a relational database table that provides a link between data in two tables. It is a column (or columns) that references a column (most often the primary key) of another table.



S.NO.	PRIMARY KEY	FOREIGN KEY
1	A primary key is used to ensure data in the specific column is unique.	A foreign key is a column or group of columns in a relational database table that provides a link between data in two tables.
2	It uniquely identifies a record in the relational database table.	It refers to the field in a table which is the primary key of another table.
3	Only one primary key is allowed in a table.	Whereas more than one foreign key are allowed in a table.
4	It is a combination of UNIQUE and Not Null constraints.	It can contain duplicate values and a table in a relational database.
5	It does not allow NULL values.	It can also contain NULL values.
6	Its value cannot be deleted from the parent table.	Its value can be deleted from the child table.
7	It constraint can be implicitly defined on the temporary tables.	It constraint cannot be defined on the local or global temporary tables.

ASP.NET - Validators

ASP.NET validation controls validate the user input data to ensure that useless, unauthenticated, or contradictory data don't get stored.

ASP.NET provides the following validation controls:

- RequiredFieldValidator
- RangeValidator
- CompareValidator
- RegularExpressionValidator
- CustomValidator
- ValidationSummary