MODULE-5

UNDERSTANDING DOCUMENT

BY PINAL PAMBHAR

Table of Contents

Introduction to WEB Development

- ASP.NET offers three frameworks for creating web applications: Web Forms, ASP.NET MVC, and ASP.NET Web Pages.
- All three frameworks are stable and mature, and you can create great web applications with any of them.
- No matter what framework you choose, you will get all the benefits and features of ASP.NET everywhere.
- Each framework targets a different development style.

ASP.NET Web Forms

- Web Forms are web pages built on the ASP.NET Technology.
- It executes on the server and generates output to the browser. It is compatible to any browser to any language supported by .NET common language runtime.
- It is flexible and allows us to create and add custom controls.
- It provides drag and drop server controls to the web forms.
- It also allows us to set properties, events, and methods for the controls.
- To write business logic, we can choose any .NET language like: Visual Basic or Visual C#.
- Web Forms are made up of two components: the visual portion (the ASPX file), and the code behind the form, which resides in a separate class file.

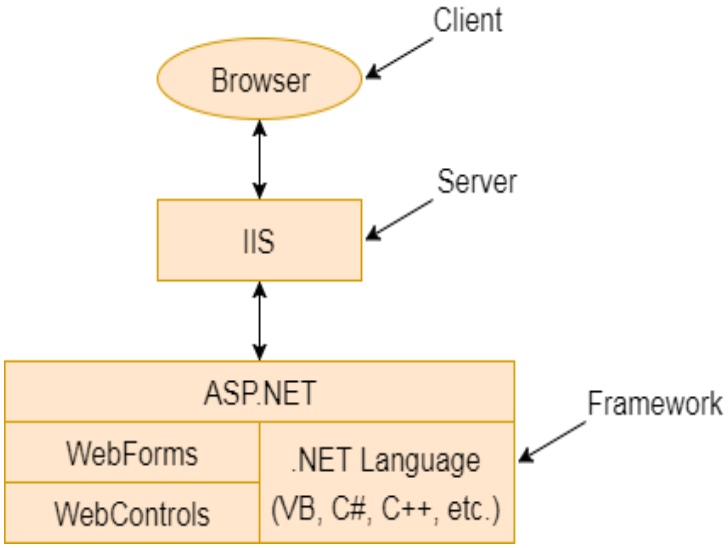


Fig: This diagram shows the components of the ASP.NET

- ASP.NET provides various Server controls.
- We have tables of all these controls below.

Control Name	Applicable Events	Description
Label	None	It is used to display text on the HTML page.
TextBox	TextChanged	It is used to create a text input in the form.
Button	Click, Command	It is used to create a button.
LinkButton	Click, Command	It is used to create a button that looks similar to the hyperlink.

ImageButton	Click	It is used to create an imagesButton. Here, an image works as a Button.
Hyperlink	None	It is used to create a hyperlink control that responds to a click event.
DropDownList	SelectedIndexChanged	It is used to create a dropdown list control.
ListBox	SelectedIndexCnhaged	It is used to create a ListBox control like the HTML control.
DataGrid	CancelCommand, EditCommand, DeleteCommand, ItemCommand, SelectedIndexChanged, PageIndexChanged, SortCommand, UpdateCommand, ItemCreated, ItemDataBound	It used to create a frid that is used to show data. We can also perform paging, sorting, and formatting very easily with this control.
DataList	CancelCommand, EditCommand, DeleteCommand, ItemCommand, SelectedIndexChanged, UpdateCommand,	It is used to create datalist that is non-tabular and used to show data.

	ItemCreated, ItemDataBound	
Repeater	ItemCommand, ItemCreated, ItemDataBound	It allows us to create a non-tabular type of format for data. You can bind the data to template items, which are like bits of HTML put together in a specific repeating format.
CheckBox	CheckChanged	It is used to create checkbox.
CheckBoxList	SelectedIndexChanged	It is used to create a group of check boxes that all work together.
RadioButton	CheckChanged	It is used to create radio button.
RadioButtonList	SelectedIndexChanged	It is used to create a group of radio button controls that all work together.
Image	None	It is used to show image within the page.
Panel	None	It is used to create a panel that works as a container.
PlaceHolder	None	It is used to set placeholder for the control.

Calendar	SelectionChanged, VisibleMonthChanged, DayRender	It is used to create a calendar. We can set the default date, move forward and backward etc.
AdRotator	AdCreated	It allows us to specify a list of ads to display. Each time the user re-displays the page.
Table	None	It is used to create table.
XML	None	It is used to display XML documents within the HTML.
Literal	None	It is like a label in that it displays a literal, but allows us to create new literals at runtime and place them into this control.

Features:

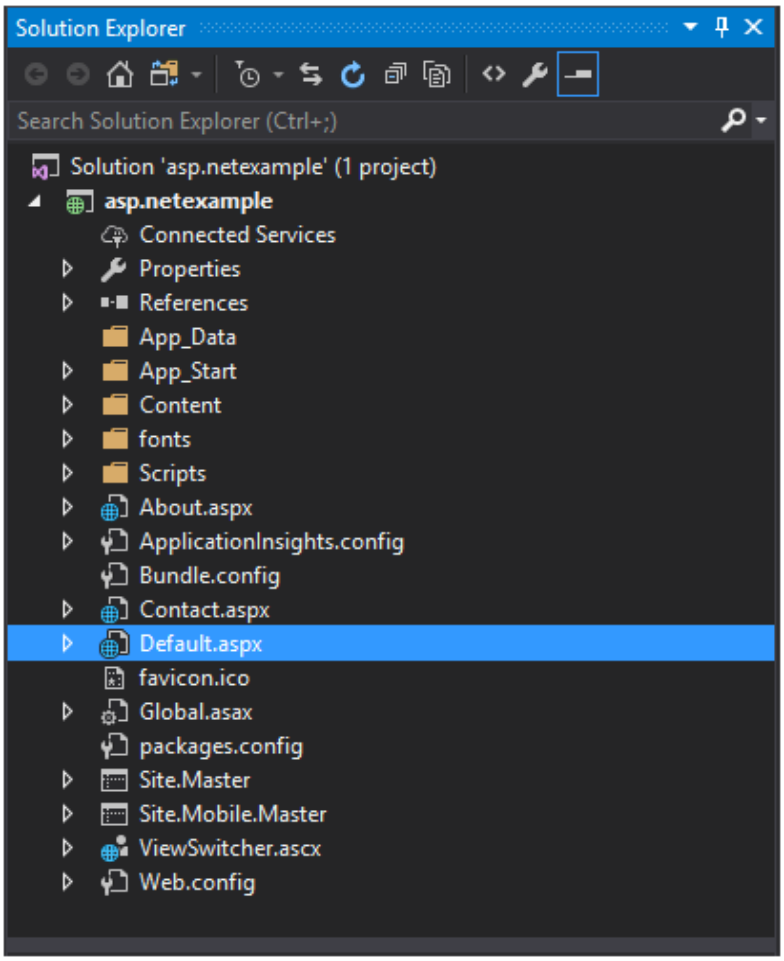
- Server Controls
- Master Pages
- Working with data
- Membership
- Client Script and Client Frameworks
- Routing
- State Management
- Security
- Performance
- Error Handling

Creating a new Web Forms project

We are using Visual studio 2019 to create web project. It includes the following steps:

1. Open Visual studio 2019 and Click on "**Create a new Project**" OR Click on the file menu from the menu bar and select **new -> project**.
2. It provides couple of choices, but we are selecting ASP.NET Web Application.
3. Now, provide **project name** and **location** to store a project and click on Create.
4. After selecting creating project, now, it asks for the type of template that we want to implement in our application.
5. Here, we are selecting **Web Forms** as because we are creating a Web Forms application.

After clicking ok, it shows project in solution explorer window that looks like the below.



MVC (Model, View, Controller)

The MVC (Model-View-Controller) is an application development pattern or design pattern which separates an application into three main components:

1. Model
2. View
3. Controller

Model

- Model is a part of the application which implements the logic for the data domain of the application.
- It is used to retrieve and store model state in a database such as SQL Server database.
- It also used for business logic separation from the data in the application.

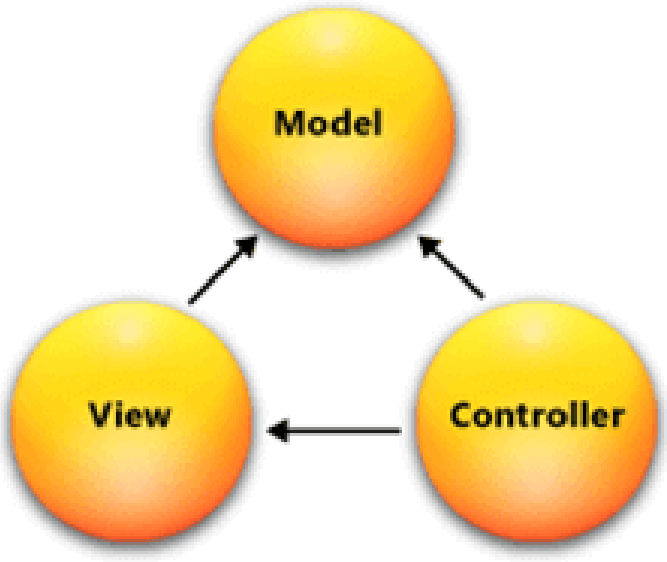
View

- View is a component that forms the application's user interface.
- It is uses to create web pages for the application.
- An example would be an edit view of a Products table that displays text boxes, drop-down lists and check boxes based on the current state of a Product object.

Controller

- Controller is the component which handles user interaction.
- It works with the model and selects the view to render the web page.
- In an MVC application, the view only displays information whereas the controller handles and responds to the user input and requests.

The following image represents the ASP.NET MVC Design Pattern:



- This design pattern is a lightweight framework which is integrated with various features such as master pages and membership-based authentication.
- It is defined in the System.Web.Mvc assembly.

Advantages of ASP.NET MVC Framework

- It manages application complexity by dividing an application into the model, view, and controller.
- It does not use view state or server-based forms. This makes the MVC framework ideal for developers who want full control over the behavior of an application.
- It provides better support for test-driven development.
- It is suitable for large scale developer team and web applications.
- It provides high degree of control to the developer over the application behavior.

Create ASP.NET MVC Application

1. Open Visual studio 2019 and Click on "**Create a new Project**" OR Click on the file menu from the menu bar and select **new -> project**.
2. It provides couple of choices, but we are selecting ASP.NET Web Application.
3. Now, provide **project name** and **location** to store a project and click on Create.
4. After selecting creating project, now, it asks for the type of template that we want to implement in our application.
5. Here, we are selecting **MVC** as because we are creating a MVC Web application.

- The ASP.NET Web API is an extensible framework for building HTTP based services that can be accessed in different applications on different platforms such as web, windows, mobile etc.
- It works more or less the same way as ASP.NET MVC web application except that it sends data as a response instead of html view.
- It is like a webservice or WCF service but the exception is that it only supports HTTP protocol.

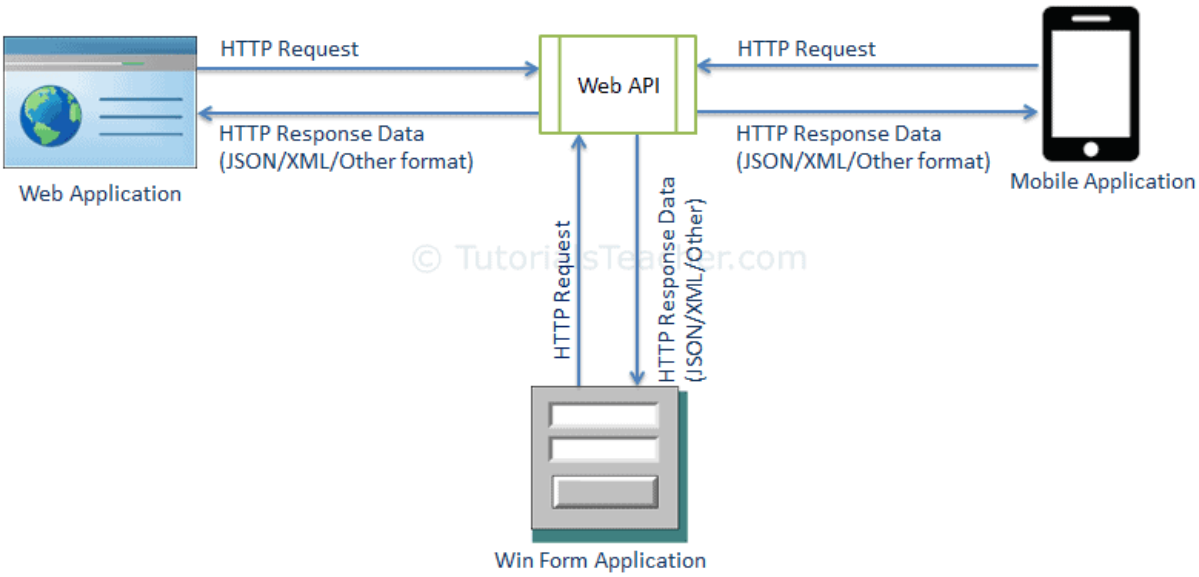


Fig. Web API

ASP.NET Web API Characteristics

- ASP.NET Web API is an ideal platform for building RESTful services.
- ASP.NET Web API is built on top of ASP.NET and supports ASP.NET request/response pipeline
- ASP.NET Web API maps HTTP verbs to method names.
- ASP.NET Web API supports different formats of response data. Built-in support for JSON, XML, BSON format.
- ASP.NET Web API can be hosted in IIS, Self-hosted or another web server that supports .NET 4.0+.

- ASP.NET Web API framework includes new HttpClient to communicate with Web API server. HttpClient can be used in ASP.MVC server side, Windows Form application, Console application or other apps.

When to choose ASP.NET Web API?

- Choose Web API if you are using .NET framework 4.0 or above.
- Choose Web API if you want to build a service that supports only HTTP protocol.
- Choose Web API to build RESTful HTTP based services.
- Choose Web API if you are familiar with ASP.NET MVC.

Start with Project

Create New Web API Project

1. Open Visual studio 2019 and Click on "**Create a new Project**" OR Click on the file menu from the menu bar and select **new -> project**.
2. It provides couple of choices, but we are selecting ASP.NET Web Application.
3. Now, provide **project name** and **location** to store a project and click on Create.
4. After selecting creating project, now, it asks for the type of template that we want to implement in our application.
5. Here, we are selecting **Web API** as because we are creating a MVC Web API.

Create Controller, Model

Create Controller:

1. In the Visual Studio, right click on the Controller **folder -> select Add -> click on Controller**.
2. This opens Add Scaffold dialog.

Note: Scaffolding is an automatic code generation framework for ASP.NET web applications. Scaffolding reduces the time taken to develop a controller, view, etc. in the MVC framework. You can develop a customized scaffolding template using templates as per your architecture and coding standards.

3. Choose Template and click **Add** button.
4. It will open the Add Controller dialog
5. In the Add Controller dialog, enter the name of the controller. Remember, the controller name must end with Controller, And click **Add**.

Create Model:

1. In the Solution Explorer.
2. Right-click on the "Model" folder then select **"Add" -> "Class"**.
3. In the Template Window, select "Installed template" -> "Visual C#" -> "Class".
4. Enter Class Name.
5. Click on the "OK" button.

Parameters Binding

- Action methods in Web API controllers can have one or more parameters of different types.
- It can be either primitive type or complex type.
- Web API binds action method parameters with the URL's query string or with the request body depending on the parameter type.
- By default, if the parameter type is of .NET primitive types such as int, bool, etc. that can be converted from string type, then it sets the value of a parameter from the query string.
- And if the parameter type is the complex type, then Web API tries to get the value from the request body by default.

The following table lists the default rules for parameter binding.

HTTP Method	Query String	Request Body
GET	Primitive Type, Complex Type	NA
POST	Primitive Type	Complex Type
PUT	Primitive Type	Complex Type
PATCH	Primitive Type	Complex Type
DELETE	Primitive Type, Complex Type	NA

- ASP.NET Web API gets the value of a primitive parameter from the query string and a complex type of parameter from the request body. But what if we want to change this default behavior?
- Use [FromUri] attribute to force Web API to get the value of complex type from the query string and [FromBody] attribute to get the value of primitive type from the request body, opposite to the default rules.

- For example, consider the following GET method.

Example: FormUri

```
public class StudentController : ApiController
{
    public Student Get([FromUri] Student stud)
    {

    }
}
```

- In the same way, consider the following example of Post() method.

Example: FormUri

```
public class StudentController : ApiController
{
    public Student Post([FromUri]Student stud)
    {

    }
}
```

- In the same way, apply the [FromBody] attribute to get the value of primitive data type from the request body instead of a query string, as shown below.

Example: FromBody

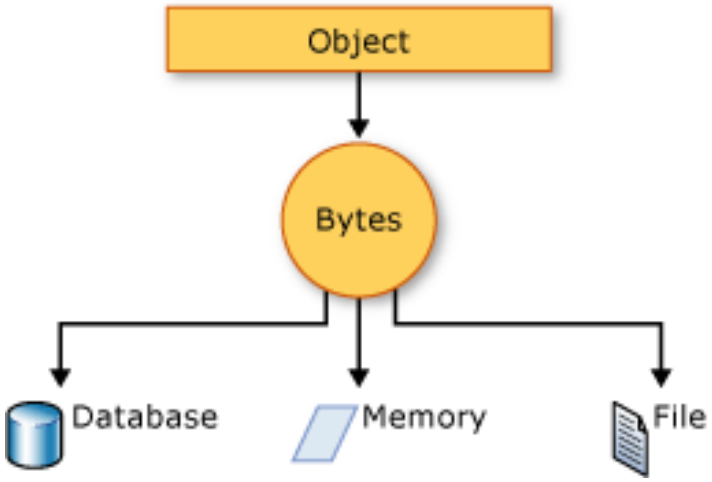
```
public class StudentController : ApiController
{
```

```
    public Student Post([FromBody]string name)
    {

    }
}
```

Serialization

- Serialization is the process of converting an object into a stream of bytes to store the object or transmit it to memory, a database, or a file.
- Its main purpose is to save the state of an object to be able to recreate it when needed.
- The reverse process is called deserialization.
- This illustration shows the overall process of serialization:



- The object is serialized to a stream that carries the data.
- The stream may also have information about the object's type, such as its version, culture, and assembly name.
- From that stream, the object can be stored in a database, a file, or memory.

Uses for serialization

- Serialization allows the developer to save the state of an object and re-create it as needed, providing storage of objects as well as data exchange.
- Through serialization, a developer can perform actions such as:

1. Sending the object to a remote application by using a web service
2. Passing an object from one domain to another
3. Passing an object through a firewall as a JSON or XML string
4. Maintaining security or user-specific information across applications

Routing

- It routes an incoming HTTP request to a particular action method on a Web API controller.
- Web API supports two types of routing:

1. Convention-based Routing
2. Attribute Routing

Convention-based Routing

- In the convention-based routing, Web API uses route templates to determine which controller and action method to execute.
- At least one route template must be added into route table in order to handle various HTTP requests.
- you can configure multiple routes in the Web API using HttpConfiguration object.

Attribute Routing

- Attribute routing is supported in Web API 2. As the name implies, attribute routing uses [Route()] attribute to define routes.
- The Route attribute can be applied on any controller or action method.
- In order to use attribute routing with Web API, it must be enabled in WebApiConfig by calling config.MapHttpAttributeRoutes() method.

Configure Web API

- Web API supports code-based configuration.
- It cannot be configured in web.config file.
- We can configure Web API to customize the behaviour of Web API hosting infrastructure and components such as routes, formatters, filters, DependencyResolver, MessageHandlers, ParamterBindingRules, properties, services etc.
- Web API configuration process starts when the application starts.
- It calls GlobalConfiguration.Configure(WebApiConfig.Register) in the Application_Start method.
- The Configure() method requires the callback method where Web API has been configured in code.
- By default this is the static WebApiConfig.Register() method.