

ASP.NET MVC

Yogesh Yadav

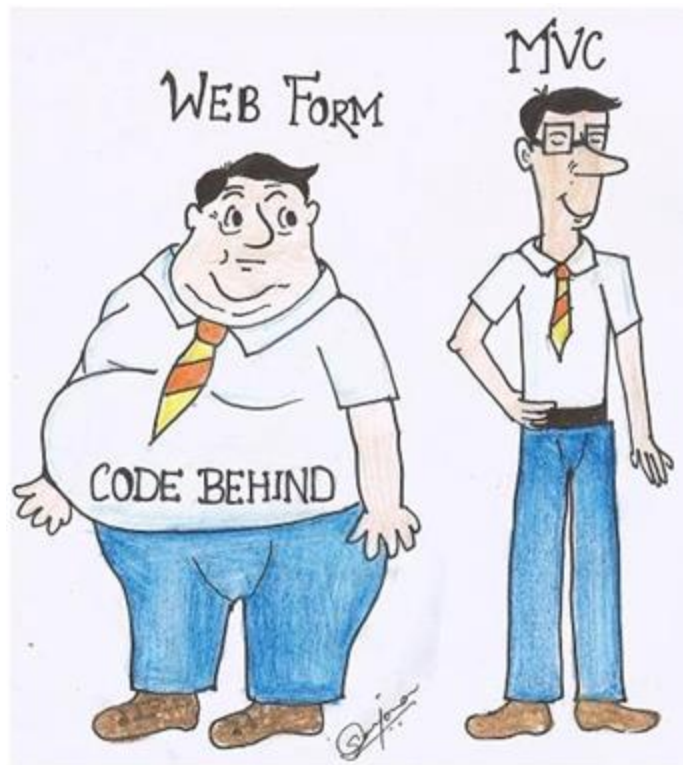
What is MVC?

- The MVC pattern is 30+ years old!
- It is a powerful and elegant means of separating concerns
- It makes it easier to test application
- It promotes parallel development thanks to the loose coupling between the three main components

MVC on the web today

- Ruby on Rails
- Django and Python
- Spring, Struts and Java
- Zend Framework and PHP
- MonoRail
- ...

Comparison with ASP.NET web forms



What web forms does well

- Represent a Page as control tree
- Give these server-side controls events like their desktop counterparts
- Hide as much HTTP and HTML as is reasonable
- Make state management as transparent as possible

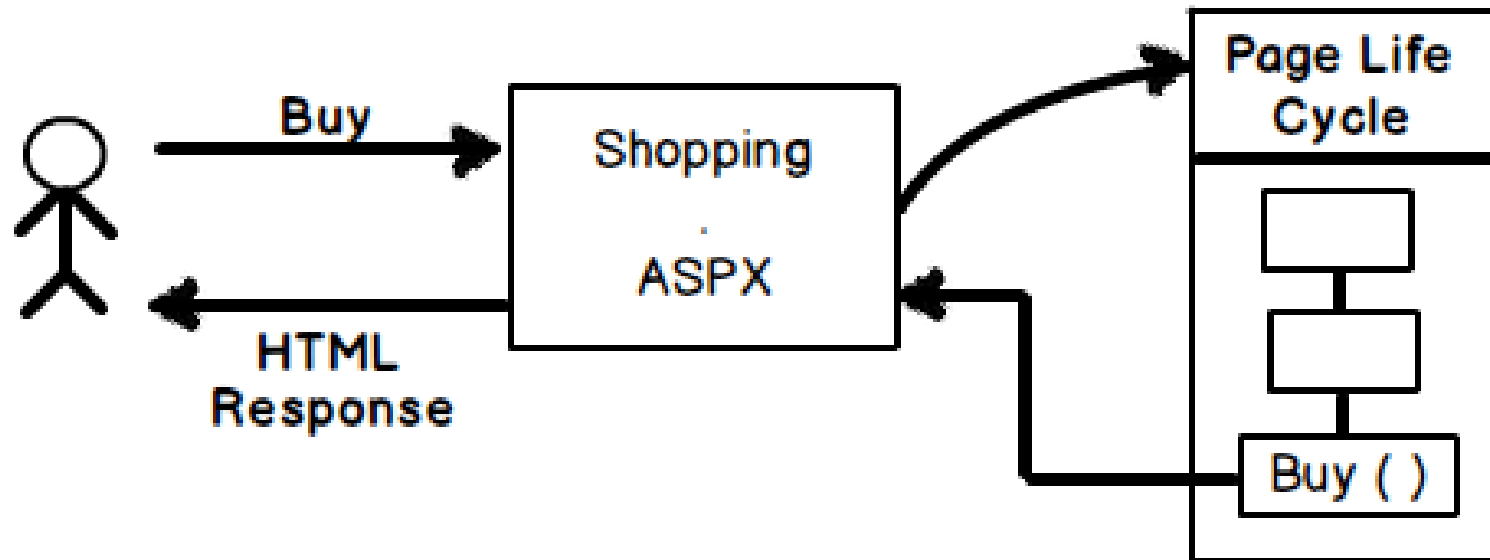
Where web forms doesn't fit

- ViewState is powerful, but it has its drawbacks (weight,...)
- Page life cycle can be a nightmare
- Limited control over HTML
- Client IDs and the

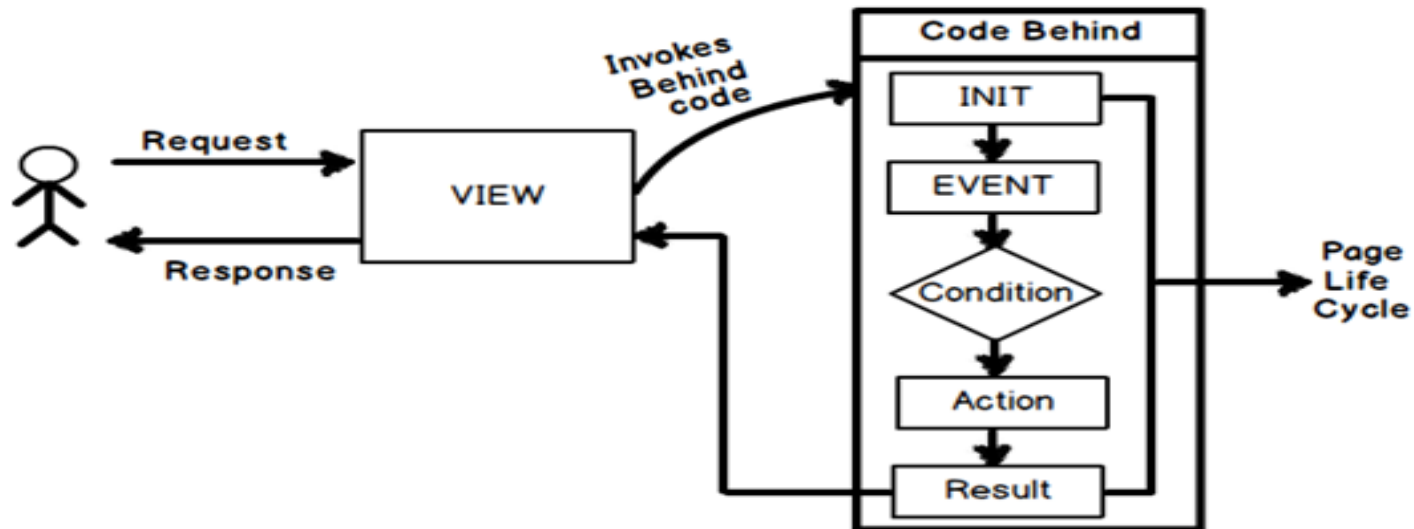
`ctl100$ContentPlaceHolder1$UserControl1$TextBox1` syndrome

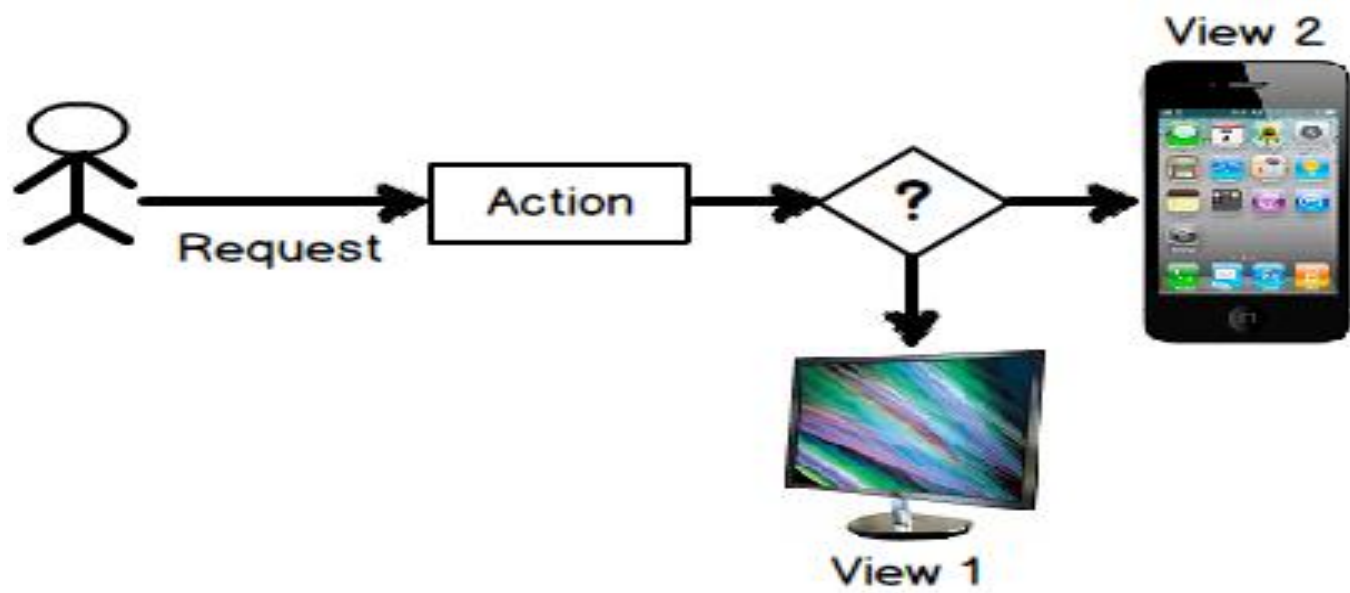
- It's nearly impossible to run a Web Form through its life cycle outside IIS

Web Form Request



View Based Request

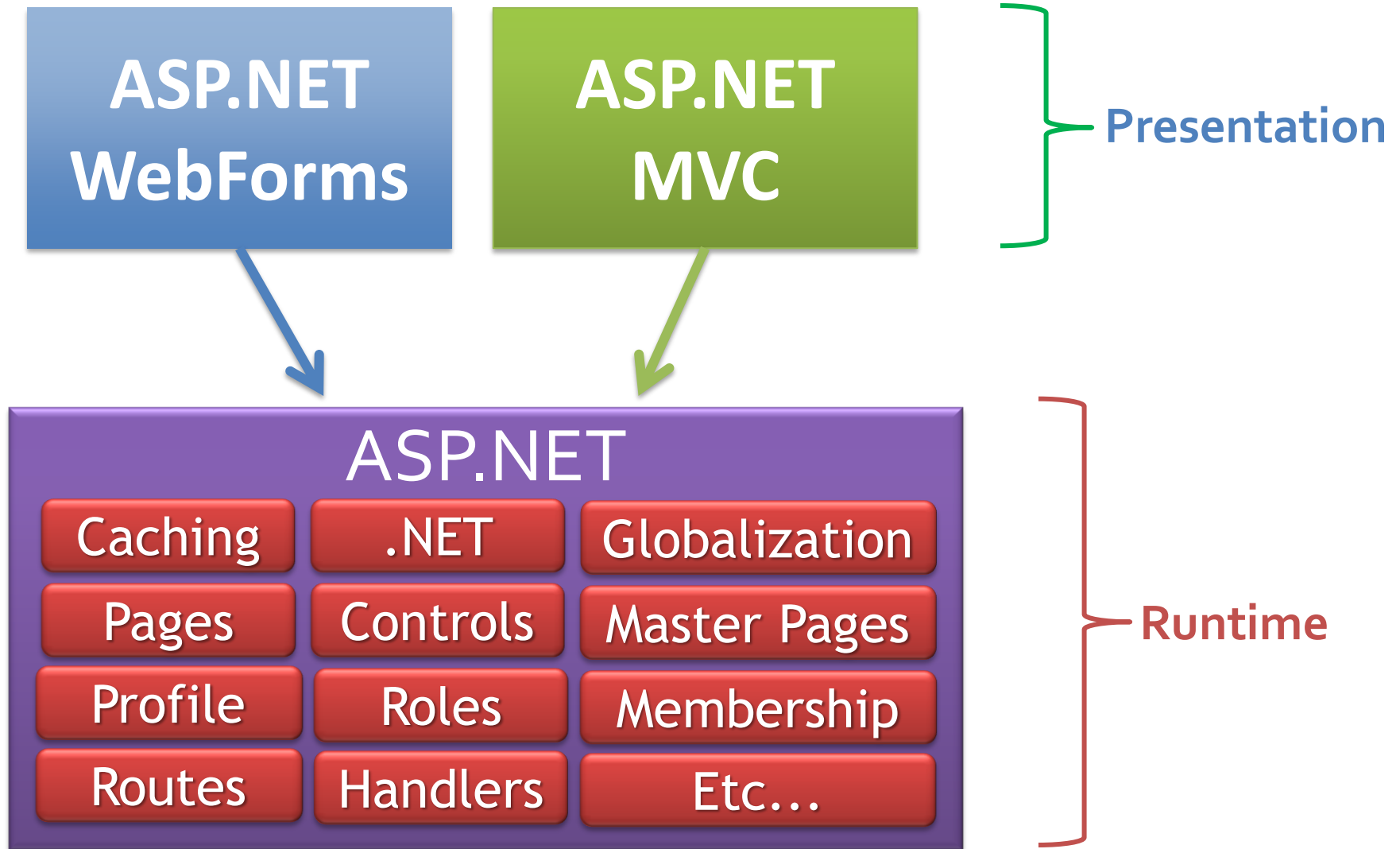




Should you fear ASP.NET MVC?

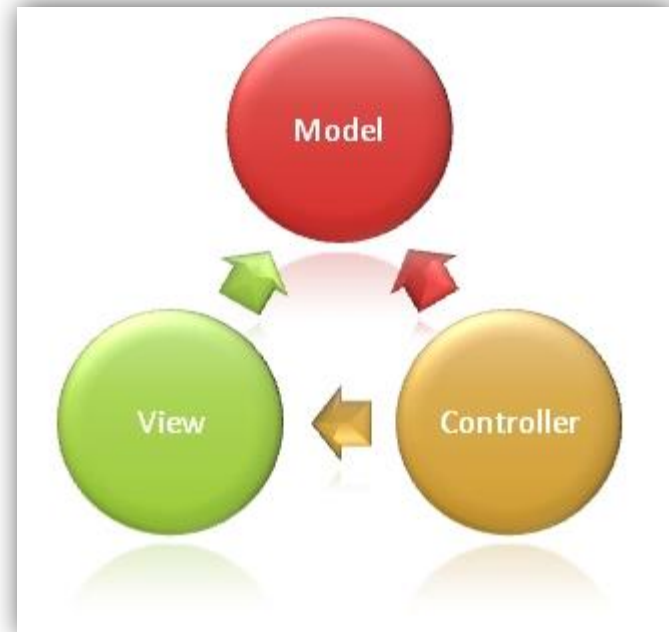
- ASP.NET MVC is built on top of ASP.NET
- ASP.NET MVC is not a replacement for anything
- It is just an alternative
- It's a totally different approach
- Embrace the web
 - User/SEO friendly URLs, HTML 5, SPA
 - Adopt REST concepts
- Uses MVC pattern
 - Conventions and Guidance
 - Separation of concerns

ASP.NET Core



The MVC pattern

- Model
 - objects are the parts of the application that implement the logic for the application's data domain
- View
 - components that display the application's user interface
- Controller
 - components that handle user interaction, work with the model, and ultimately select a view to render



Model

- Business logic and validation of the application's data domain
- Totally independant from the views or the controllers
- Model state can be stored in memory, database, XML files,...

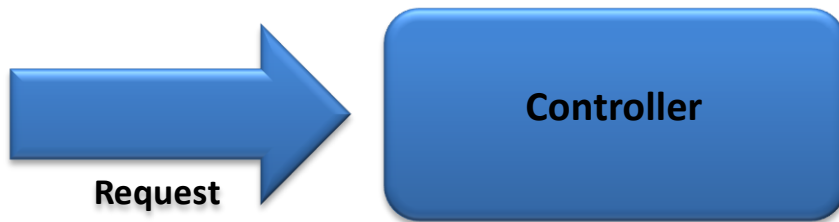
View

- Application's user interface using data from the model
- No interaction with the models or the controllers
- Views can be strongly typed
- Almost no code

Controller

- Handle user interaction
- Query the model
- Select the right view to render

MVC Flow



Step 1

Incoming request directed to **Controller**

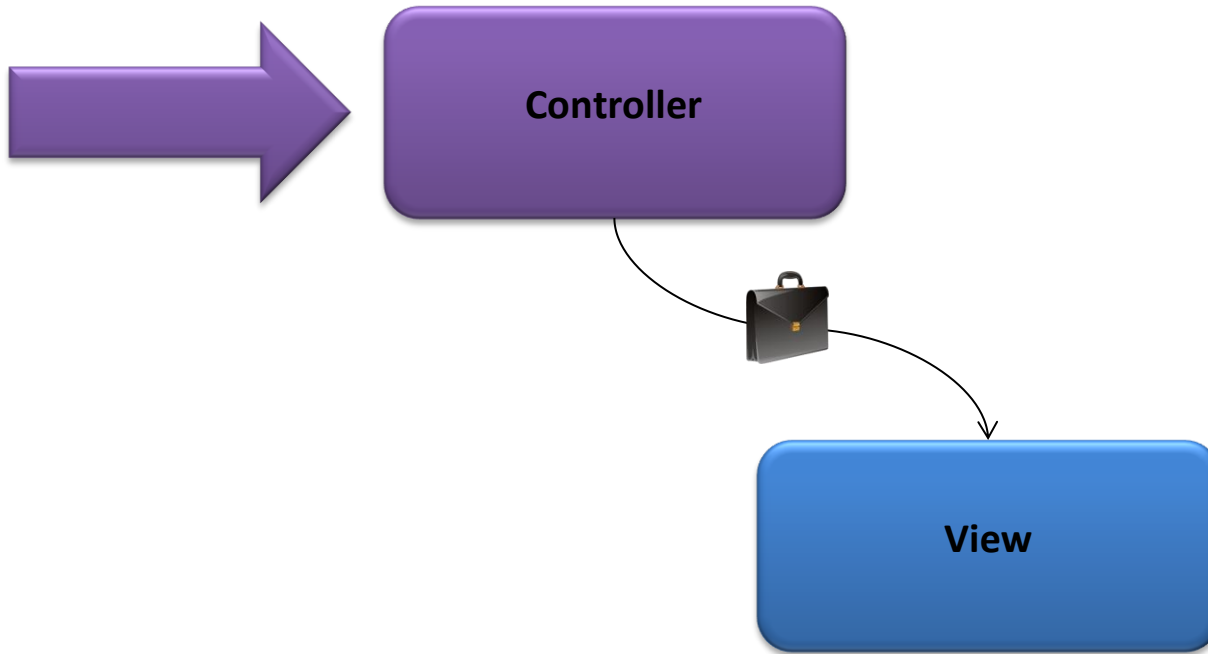
MVC Flow



Step 2

Controller processes request and forms a data **Model**

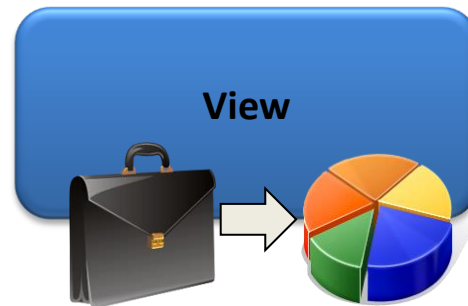
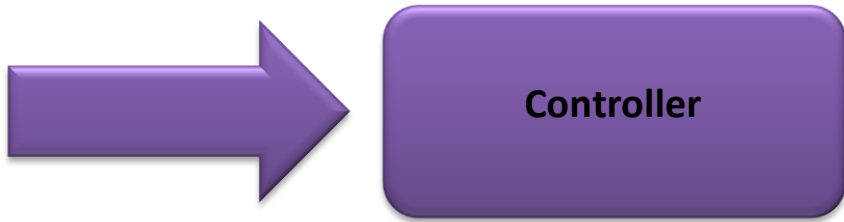
MVC Flow



Step 3

Model is passed to **View**

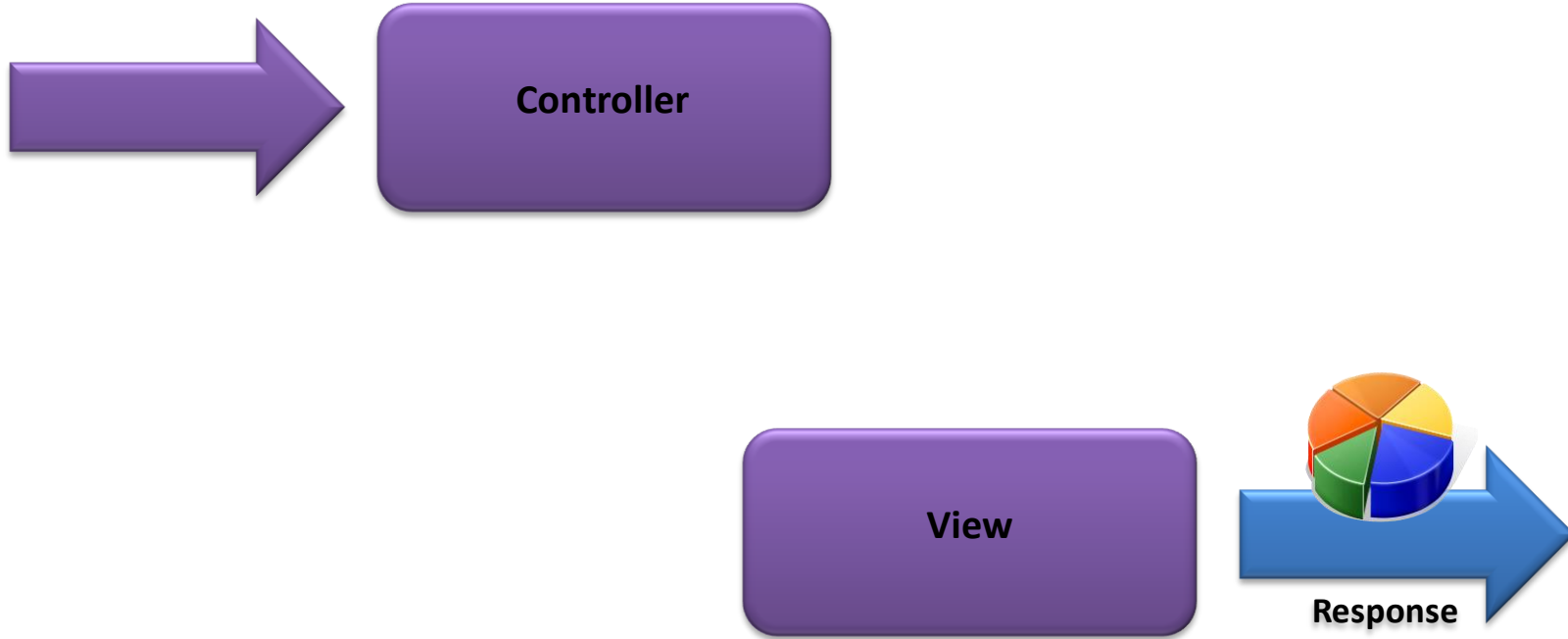
MVC Flow



Step 4

View transforms **Model** into appropriate output format

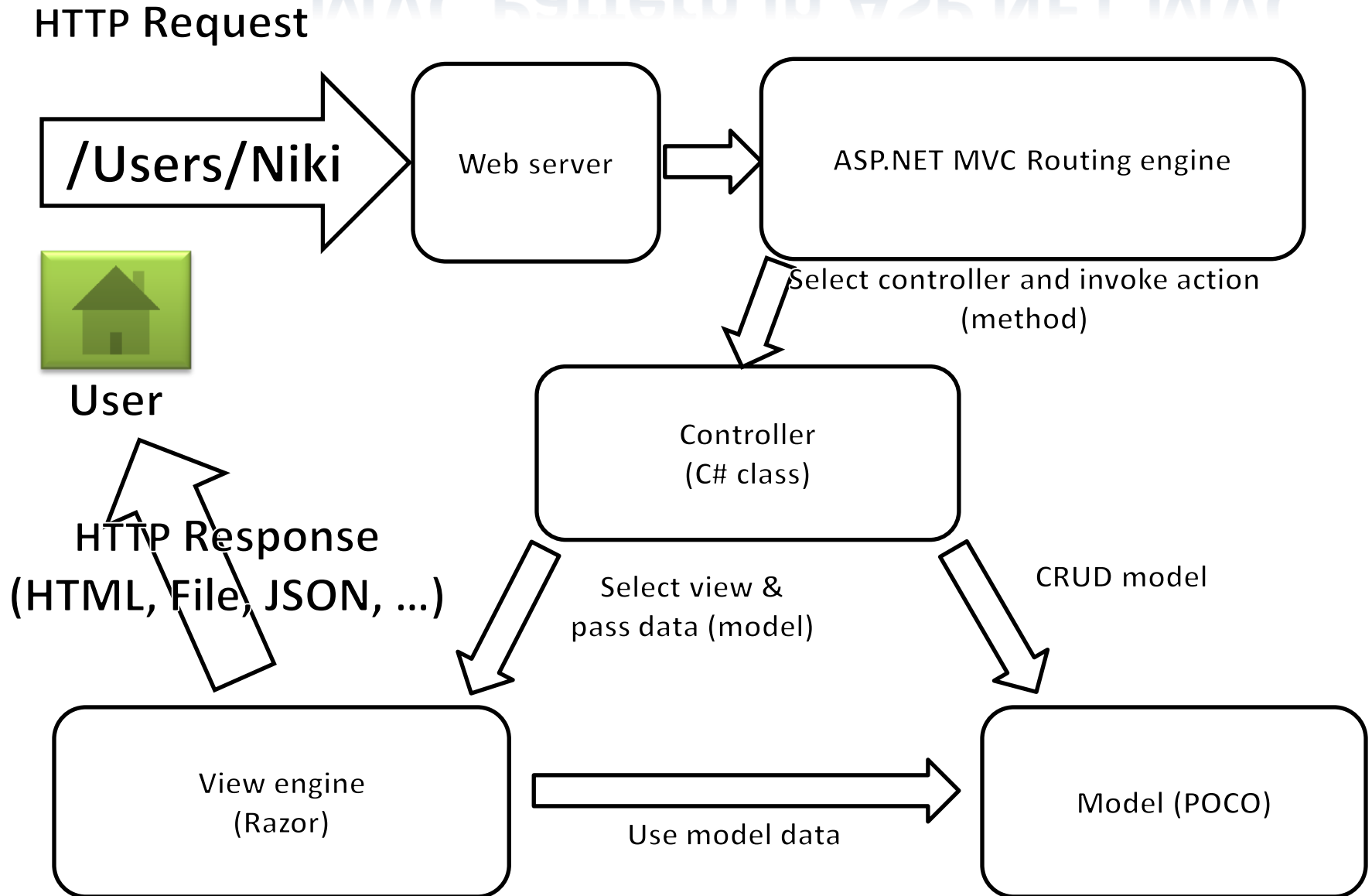
MVC Flow



Step 5

Response is rendered

MVC Pattern in ASP.NET MVC



ASP.NET MVC ROUTING



ASP.NET MVC ROUTING

- Mapping between patterns and a combination of **controller + action + parameters**
- Routes are defined as a global list of routes
 - `System.Web.Routing.RouteTable.Routes`
- Greedy algorithm
 - the first match wins



Register routes

- ◆ In Global.asax in the Application_Start() there is `RouteConfig.RegisterRoutes(RouteTable.Routes);`
- ◆ RoutesConfig class is located in /App_Start/ in internet applications template by default

```
public static void RegisterRoutes(RouteCollection routes)
{
    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

    routes.MapRoute(
        name: "Default",
        url: "{controller}/{action}/{id}",
        defaults: new {
            controller = "Home",
            action = "Index",
            id = UrlParameter.Optional
        }
    );
}
```

Routes to ignore
The [*] means all left

Route name

Route pattern

Default parameters

Routing Examples

```
public static void RegisterRoutes(RouteCollection routes)
{
    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

    routes.MapRoute(
        name: "Default",
        url: "{controller}/{action}/{id}",
        defaults: new {
            controller = "Home",
            action = "Index",
            id = UrlParameter.Optional
        }
    );
}
```

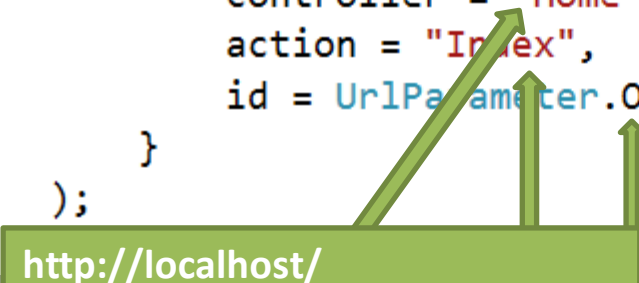
http://localhost/Products/ById/3

- ◆ Controller: Products
- ◆ Action: ById
- ◆ Id: 3

Routing Examples

```
public static void RegisterRoutes(RouteCollection routes)
{
    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

    routes.MapRoute(
        name: "Default",
        url: "{controller}/{action}/{id}",
        defaults: new {
            controller = "Home",
            action = "Index",
            id = UrlParameter.Optional
        }
    );
}
```



- ◆ Controller: Home
- ◆ Action: Index
- ◆ Id: 0 (optional parameter)

Custom Route

```
routes.MapRoute(  
    name: "Users",  
    url: "Users/{username}",  
    defaults: new  
    {  
        controller = "Users",  
        action = "ByUsername",  
    }  
);
```

http://localhost/Users

?

◆ Result: 404 Not Found

Route Constraints

- ◆ Constraints are rules on the URL segments
- ◆ All the constraints are regular expression compatible with class `Regex`
- ◆ Defined as one of the `routes.MapRoute(...)` parameters

```
// 2013/01/29/Blog-title
routes.MapRoute(
    name: "Blog",
    url: "{year}/{month}/{day}",
    defaults: new { controller = "Blog", action = "ByDate" },
    constraints: new { year=@"\d{4}", month=@"\d{2}", day=@"\d{2}" }
);
```

```
constraints: new { year=@"\d{4}", month=@"\d{2}", day=@"\d{2}" }
```

MVC Application

Server Project

↑ ↑

<http://localhost/MVCDemo/Home/Index>

```
public class HomeController : Controller
{
    public string Index()
    {
        return "Hello from MVC Application";
    }
}
```

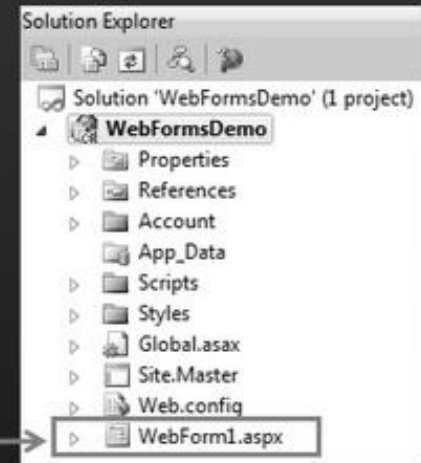
In MVC URL's are mapped to controller Action Methods

In a WebForms URL's are mapped to Physical Files

Server Project

↑ ↑

<http://localhost/WebFormsDemo/WebForm1.aspx>



Please Note: Functions in a controller are generally called as Controller Action Methods

Controller in MVC Application

Server Project

↑ ↑

<http://localhost/MVCDemo/Home/Index>

```
public class HomeController : Controller
{
    public string Index()
    {
        return "Hello from MVC Application";
    }
}
```

So, how are the URL's mapped to Controller Action Methods?

The answer is **ASP.NET Routing**. Notice that in **Global.asax** we have **RegisterRoutes()**.

```
protected void Application_Start()
{
    AreaRegistration.RegisterAllAreas();

    WebApiConfig.
        Register(GlobalConfiguration.Configuration);
    FilterConfig.
        RegisterGlobalFilters(GlobalFilters.Filters);
    RouteConfig.RegisterRoutes(RouteTable.Routes);
}
```

<http://localhost/MVCDemo/home/index/10?name=Pragim>

ASP.NET MVC will automatically pass any query string or form post parameters named "name" to Index action method when it is invoked

```
public string Index(string id, string name)
{
    return "The value of Id = " + id + " and Name = " + name;
}
```

```
public string Index(string id)
{
    return "The value of Id = " + id + " and Name = " + Request.QueryString["name"];
}
```

Controller

A controller will decide what to do and what to display in the view. It works as follows:

- i) A request will be received by the controller
- ii) Basing on the request parameters, it will decide the requested activities
- iii) Basing on the request parameters, it will delegates the tasks to be performed
- iv) Then it will delegate the next view to be shown

View in MVC Application

```
public ActionResult Index(string id, string name)
{
    ViewBag.Countries = new List<string>()
    {
        "India",
        "US",
        "UK",
        "Canada"
    };

    return View();
}
```

ViewBag & ViewData is a mechanism to pass data from controller to view

Please Note: To pass data from controller to a view, It's always a good practice to use strongly typed view models

We use "@" symbol to switch between html and C# code

```
@{
    ViewBag.Title = "Countries List";
}

<h2>Countries List</h2>
<ul>
    @foreach (string strCountry in ViewBag.Countries)
    {
        <li>@strCountry</li>
    }
</ul>
```

Countries List

- India
- US
- UK
- Canada

Action Method Return

An action method is used to return an instance of any class which is derived from ActionResult class.

Some of the return types of a controller action method are:

- i) ViewResult : It is used to return a webpage from an action method
- ii) PartialViewResult : It is used to send a section of a view to be rendered inside another view.
- iii) JavaScriptResult : It is used to return JavaScript code which will be executed in the user's browser.
- iv) RedirectResult : Based on a URL, It is used to redirect to another controller and action method.
- v) ContentResult : It is an HTTP content type may be of text/plain. It is used to return a custom content type as a result of the action method.
- vi) JsonResult : It is used to return a message which is formatted as JSON.
- vii) FileResult : It is used to send binary output as the response.
- viii) EmptyResult : It returns nothing as the result.

Convention used by MVC to find views

- Notice that, the **Index()** action method does not specify the name of the view. So,
 1. How does asp.net mvc know, which view to use
 2. What locations does asp.net mvc search

- ```
public ActionResult Index()
{
 return View();
}
```

- The view 'Index' or its master was not found or no view engine supports the searched locations. The following locations were searched:

~/Views/Employee/Index.aspx

~/Views/Employee/Index.ascx

~/Views/Shared/Index.aspx

~/Views/Shared/Index.ascx

~/Views/Employee/Index.cshtml

~/Views/Employee/Index.vbhtml

~/Views/Shared/Index.cshtml

~/Views/Shared/Index.vbhtml

So, from the error message, it should be clear that, **MVC looks for a view with the same name**

- **Please note that, the view extension can be any of the following**
  - a).cshtml**
  - b).vbhtml**
  - c).aspx**
  - d).ascx**

**If I have all of the following files in "Views/Employee" folder, then MVC picks up "Index.aspx"**

- **If you want to use "Index.cshtml" instead, then specify the full path as shown below.**

```
public ActionResult Index()
{
 var employees =
db.Employees.Include("Department");
 return View("~/Views/Employee/Index.cshtml",
employees.ToList());
}
```

If you specify only the name of the view along with its extension as shown below, you will get an error.

```
return View("Index.cshtml", employees.ToList());
```

If you want to use a view name which is not inside the views folder of the current controller, then specify the full path as shown below.

# Razor view syntax

Use @ symbol to switch between c# code and html

```
@for (int i = 1; i <= 10; i++)
{
 @i
}
```

1 2 3 4 5 6 7 8 9 10

Use @{ } to define a code block. If we want to define some variables and perform calculations, then use code block.

```
@{
 int SumOfEvenNumbers = 0;
 int SumOfOddNumbers = 0;

 for(int i =1; i<=10; i++)
 {
 if(i %2 == 0)
 {
 SumOfEvenNumbers = SumOfEvenNumbers + i;
 }
 else
 {
 SumOfOddNumbers = SumOfOddNumbers + i;
 }
 }
}
```

Sum of Even Numbers = 30

Sum of Odd Numbers = 25

```
<h3>Sum of Even Numbers = @SumOfEvenNumbers</h3>
<h3>Sum of Odd Numbers = @SumOfOddNumbers</h3>
```

# Razor view syntax

Use <text> element or @: to switch between c# code and literal text

```
@for (int i = 1; i <= 10; i++)
{
 @i
 if (i % 2 == 0)
 {
 <text> - Even </text>
 }
 else
 {
 <text> - Odd </text>
 }

}
```

OR

```
@for (int i = 1; i <= 10; i++)
{
 @i
 if (i % 2 == 0)
 {
 @: - Even
 }
 else
 {
 @: - Odd
 }

}
```

1 - Odd  
2 - Even  
3 - Odd  
4 - Even  
5 - Odd  
6 - Even  
7 - Odd  
8 - Even  
9 - Odd  
10 - Even

# View Engines

- Out of the box ASP.NET offers the following 2 View engines
  - 1. ASPX
  - 2. Razor
- 1. what is the difference between Razor and ASPX view engines?
- It mostly, boils down to the syntax. otherwise there are no major differences between the two. In ASPX view engine, the server side script is wrapped between `<% %>`, where as in RAZOR we use `@`. Personally I prefer using RAZOR views, as it is very easy to switch between HTML and Code.



- 2. Is it possible, to have both RAZOR and ASPX views in one application?
- Yes, When you right click on any controller action method, and select "Add View" from the context menu, you will have the option to choose the view engine of your choice from the "Add View" dialog box.
- 3. Is it possible, to use a third party view engine with asp.net MVC?