

(420-PS4-AB)

Developing ASP .NET MVC (01)

Aref Mourtada

Fall 2017

Outline

- ASP. NET Web forms
- What is MVC?
- Development Environment.
- First MVC Web Application.
- Demos: Controller – Views
- Controller & View Relation
- Adding Themes
- ActionResult
- Action Parameters
- Custom Routes

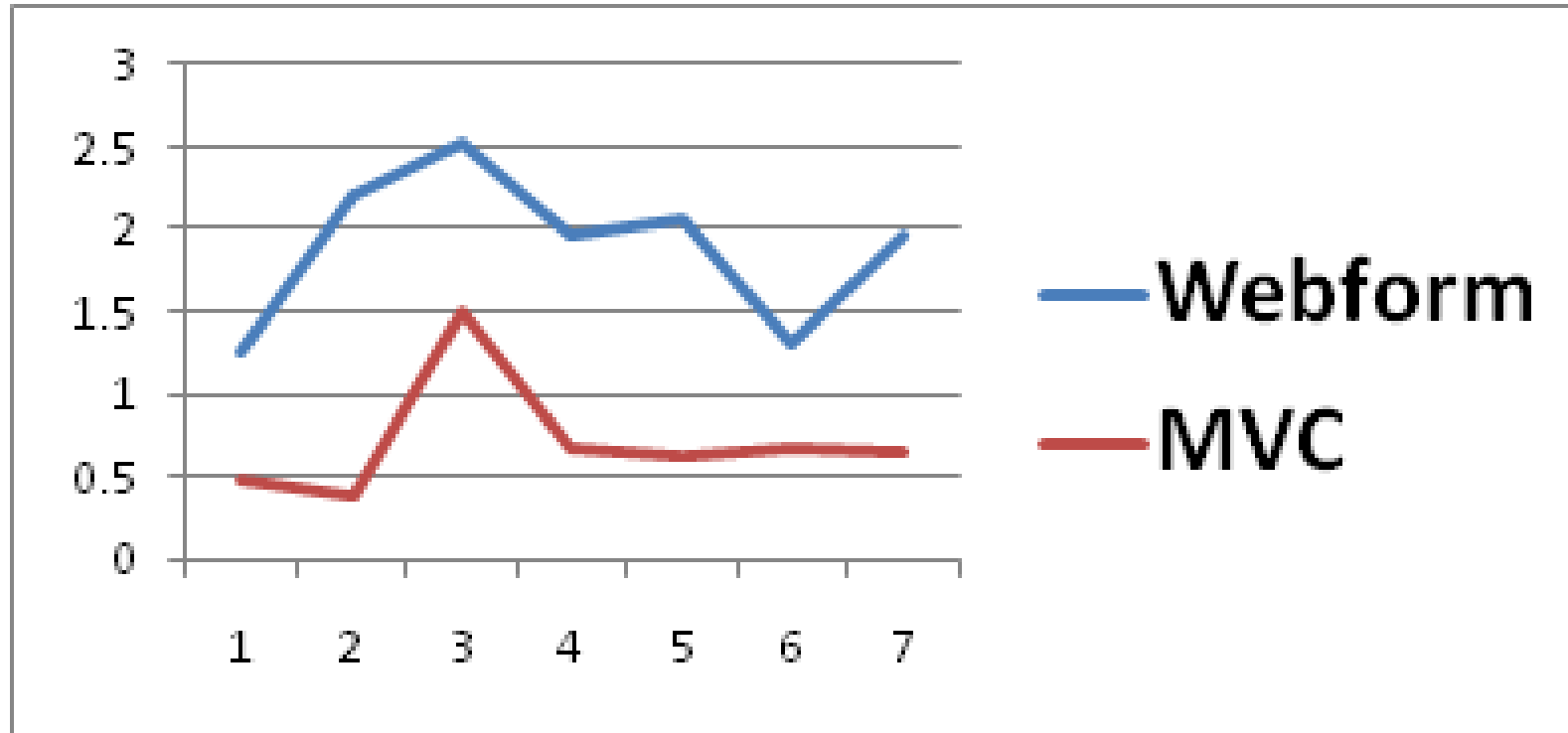
ASP .NET Web Forms

- ASP.NET Webforms has served and successfully delivered web application for past years 15 years.
 - And still is.
- Reason for success:
 - Rapid application development.
 - Visual programming approach → Visual Studio
- UI: Drag & drop → backend: code behind

Why create a new technology?

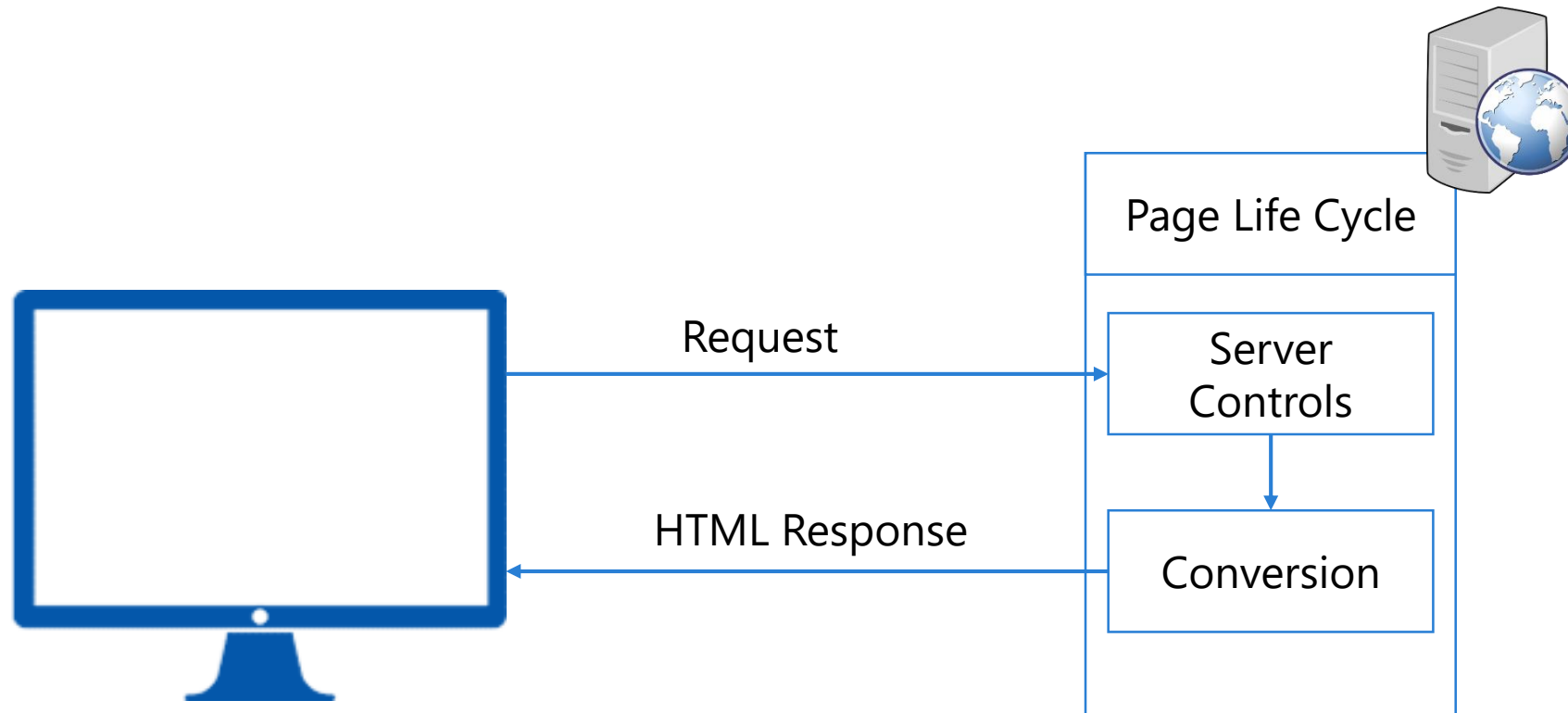
- ASP .NET was a great success.
- Bottlenecks:
 - Response time: How fast the server responds to request?
 - Bandwidth consumption: How much data is generated/sent?

Response Time

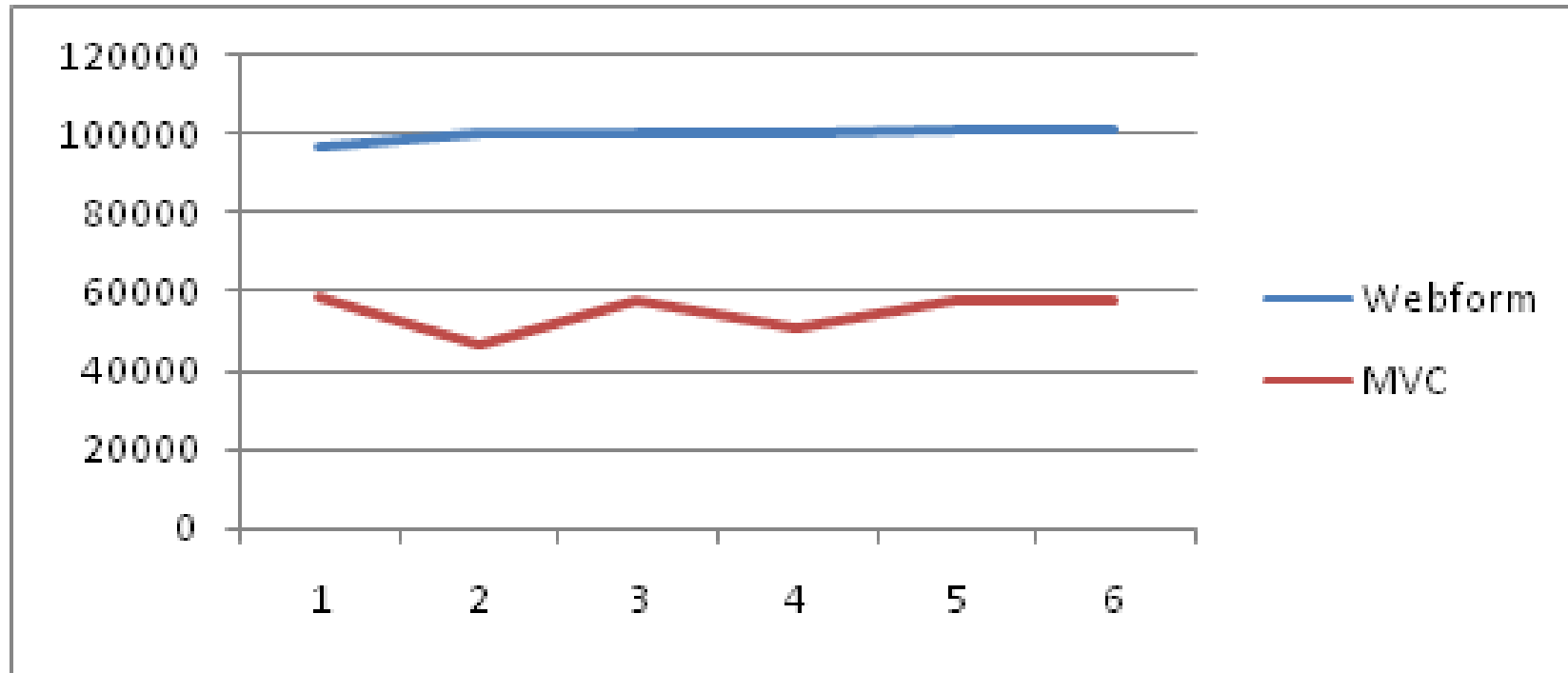


Source: <https://www.codeproject.com/Articles/864950/ASP-NET-MVC-vs-ASP-NET-WebForm-performance-compari>

APS .NET Web Forms Life Cycle

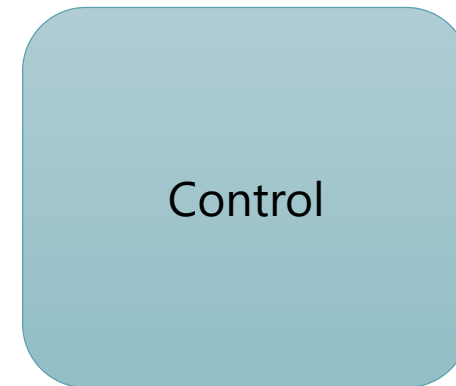
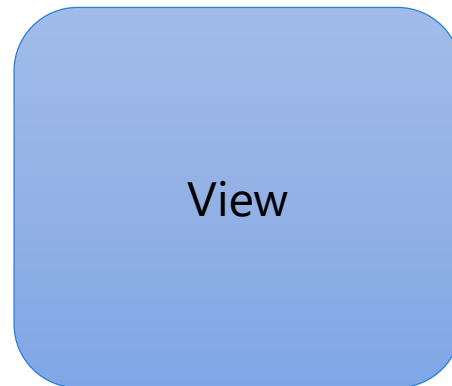
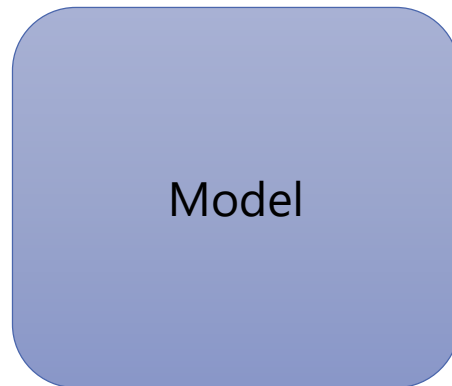


Bandwidth Consumption



Source: <https://www.codeproject.com/Articles/864950/ASP-NET-MVC-vs-ASP-NET-WebForm-performance-compari>

MVC Architecture Pattern



MVC

- Designed in 1970s for desktop applications.
- Widely adopted for web application development.
- Server frameworks are created based on MVC:
 - ASP .NET MVC
 - Ruby on Rails
 - Express

Model

- Represents application data and behavior
 - In terms of problem domain
 - Independent of UI.

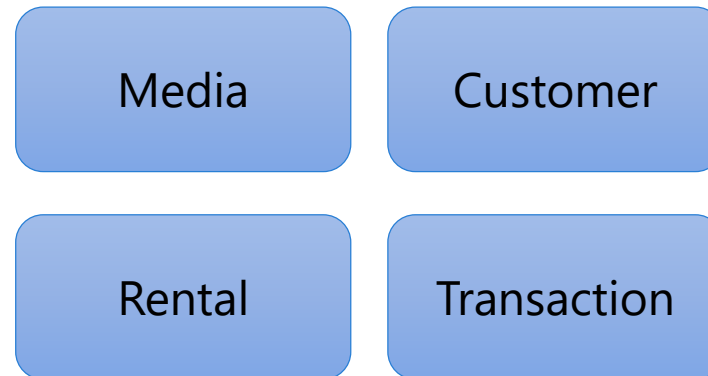


Model

Model Example

- Video Rental Web Application:

- Classes



- Properties & methods
 - Not connected to UI → can be used with any other context (desktop, mobile)
 - Plain old CLR Objects (POCOS) → No dependencies

View

Represents the HTML markup the we display to the user.



View

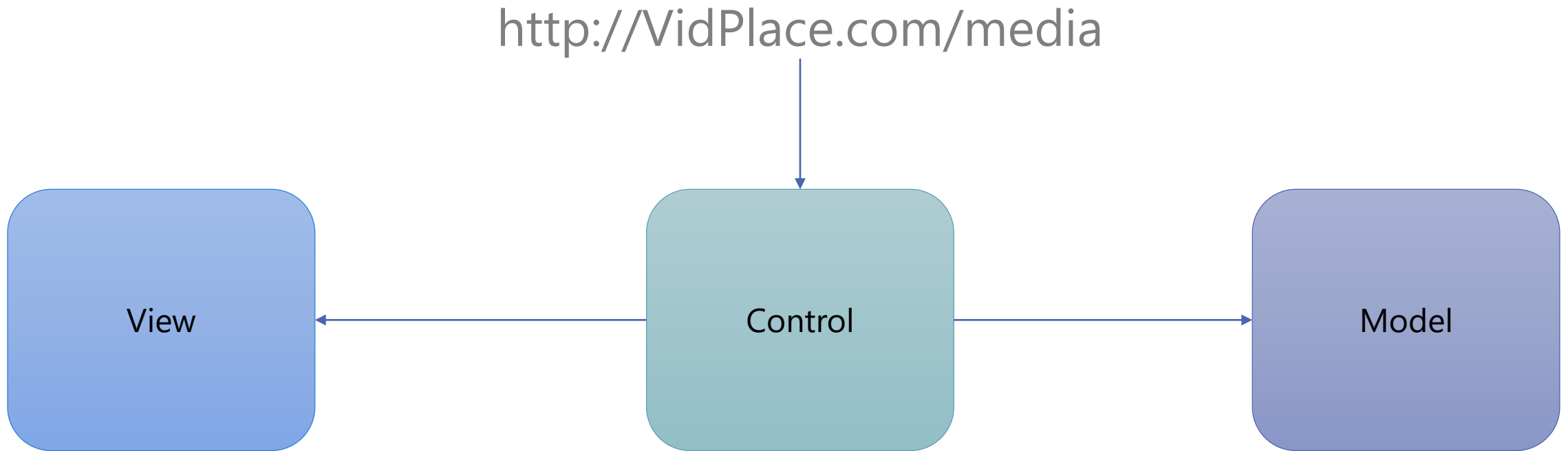
Control

Responsible for handling an HTTP Request.



Control

Example



Better separation of concerns
Better maintainable application

MVC is NOT

- **NOT** A completely new way of doing everything.
- **NOT** Too complex to learn in time for your project.
- **NOT** Someone's attempt to make your life more difficult.

Development Environment

- For MVC 5.0
 - Visual Studio 2013 or later
- Tools → Extensions and Updates → Online
 - Productivity Power Tools
 - Web Essentials
- ReSharper
 - 3rd party pluglin
 - Provides several time saving tips and tricks
 - Free 30 days version available

Demo: First MVC Application

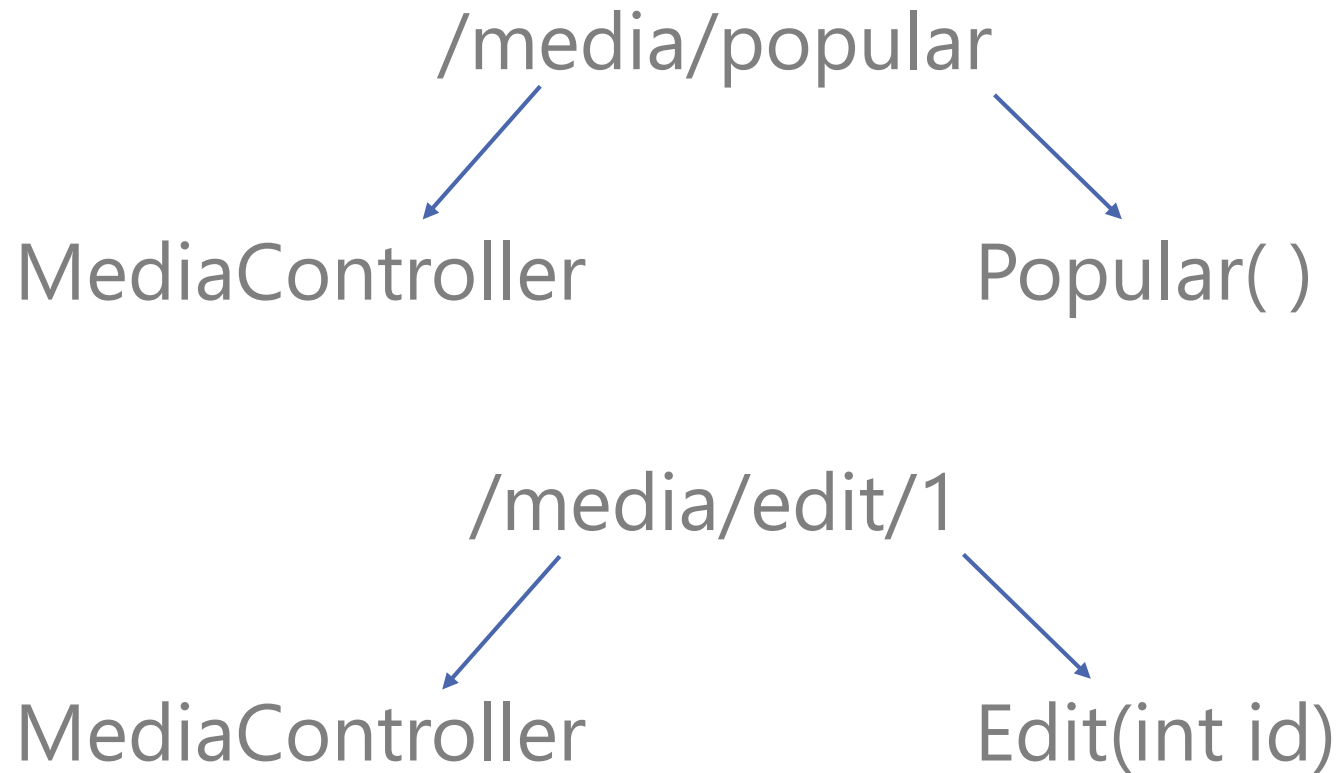
- Add new project
- ASP.NET Web Application
- Name it: VidPlace
- Select MVC

Route Configuration

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

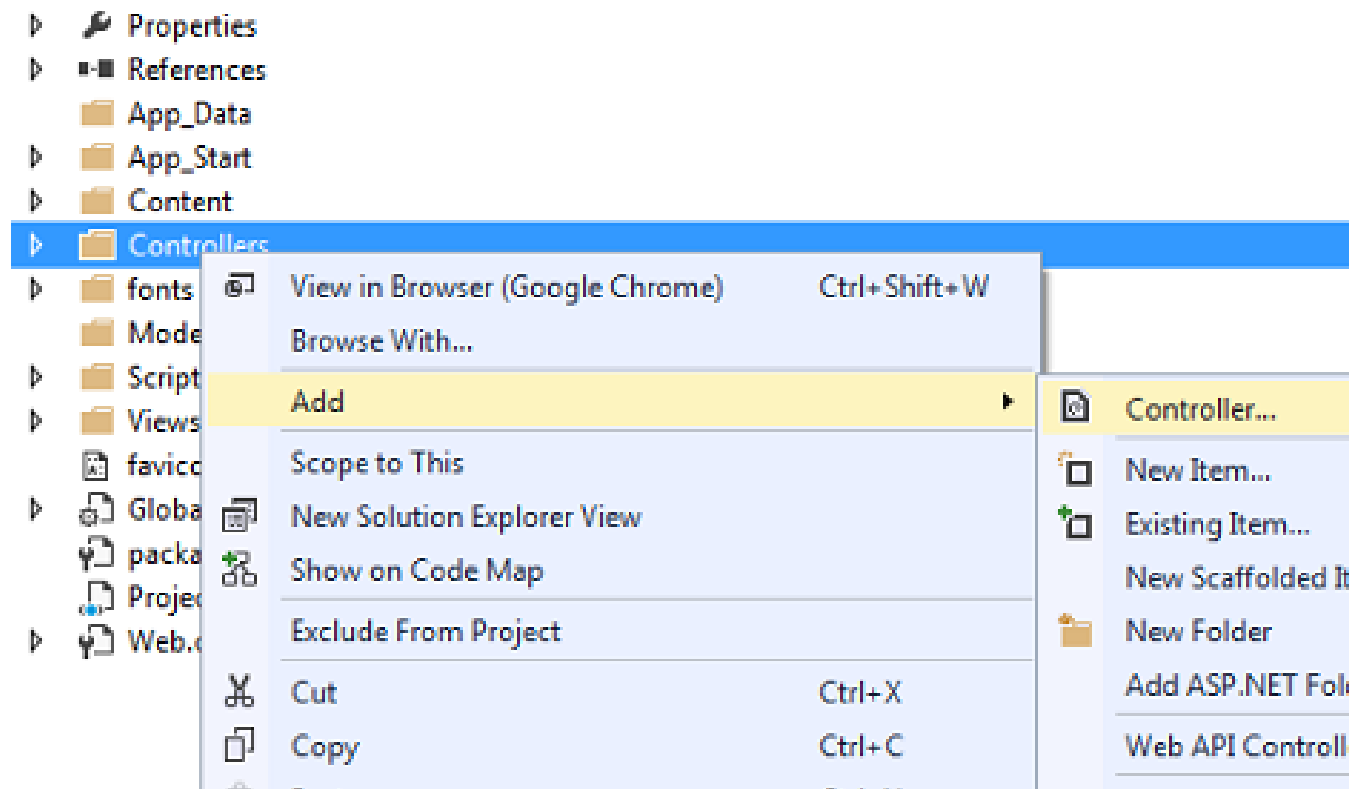
        routes.MapRoute(
            name: "Default",
            url: "{controller}/{action}/{id}",
            defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }
        );
    }
}
```

Example: Routing



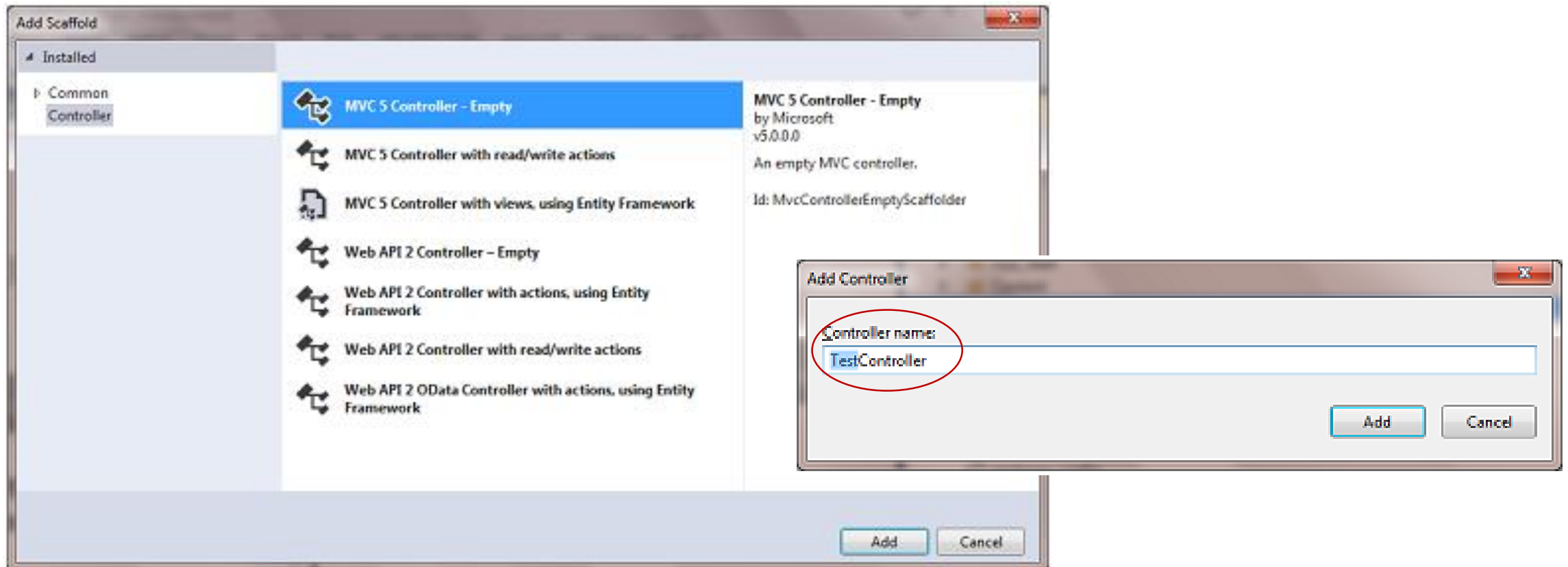
Demo: HelloWorld

- Controllers → Add → Controller



Demo

Select “MVC 5 Controller – Empty” and click Add



Demo: Controller (1)

```
public string GetString()  
{  
    return "Hello World, Welcome to MVC";  
}
```

- To navigate to page:
 - In the address bar place: "ControllerName/ActionName"

Demo: Controller (2)

```
public class Customer
{
    public string CustomerName { get; set; }
    public string Address { get; set; }
}
public class TestController : Controller
{
    public Customer GetCustomer()
    {
        Customer c = new Customer();
        c.CustomerName = "Customer 1";
        c.Address = "Address1";
        return c;
    }
}
```

Demo: Controller (3) Override toString

```
public override string ToString()  
{  
    return this.CustomerName+"|"+this.Address;  
}
```

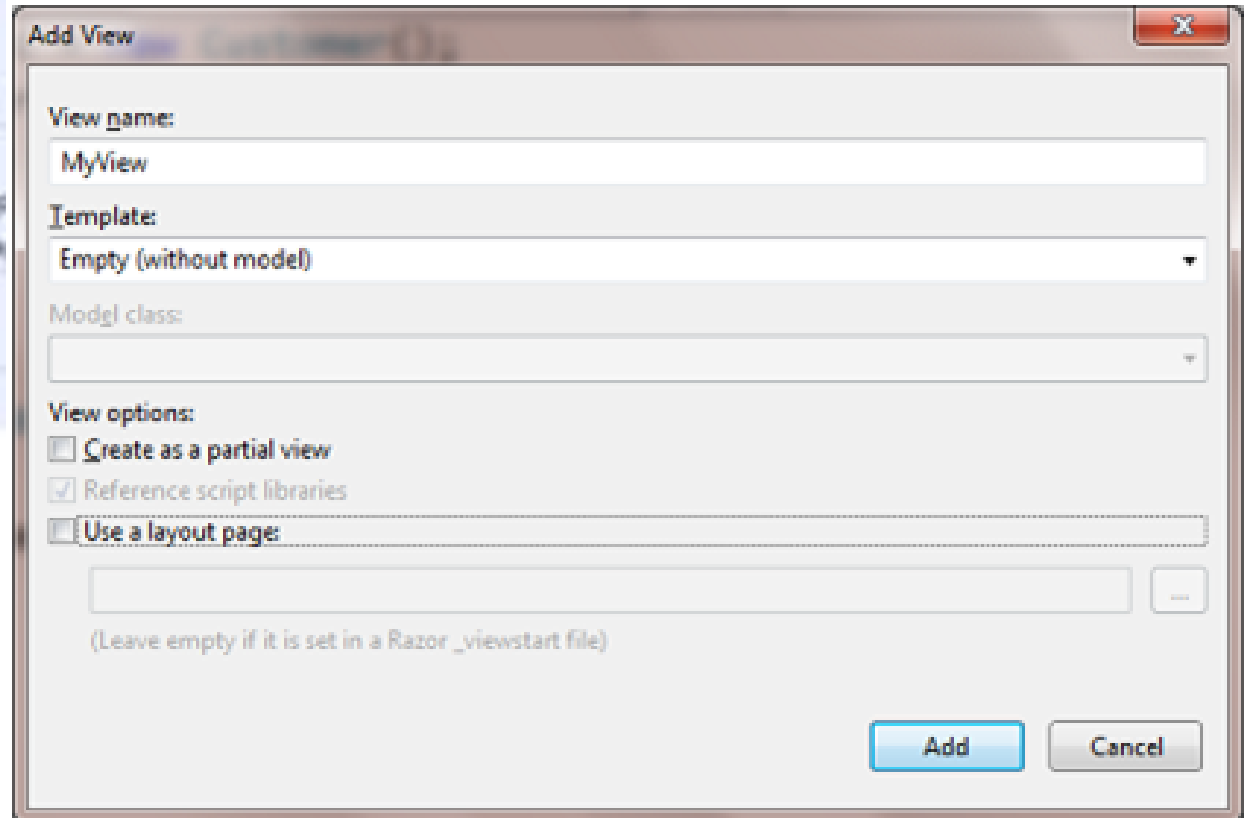
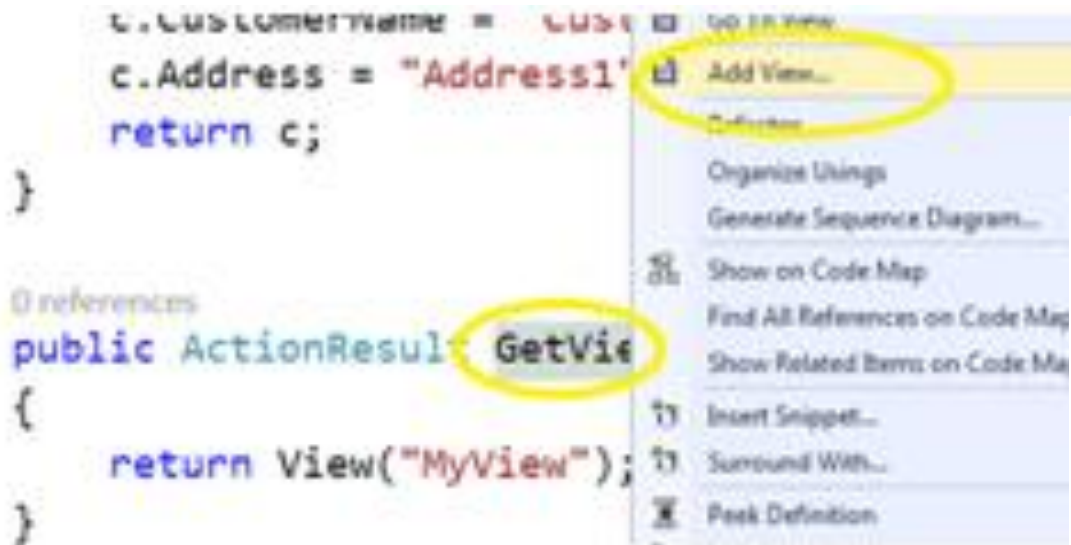

Demo: Views

- Under the TestController, add

```
public ActionResult GetView()  
{  
    return View("MyView");  
}
```

Demo: Add View

```
c.CustomerName = cus  
c.Address = "Address1"  
return c;  
}  
  
0 references  
public ActionResult GetView  
{  
    return View("MyView");  
}
```



Add View

View name:
MyView

Template:
Empty (without model)

Model class:

View options:
☐ Create as a partial view
☒ Reference script libraries
☐ Use a layout page:
(Leave empty if it is set in a Razor _viewstart file)

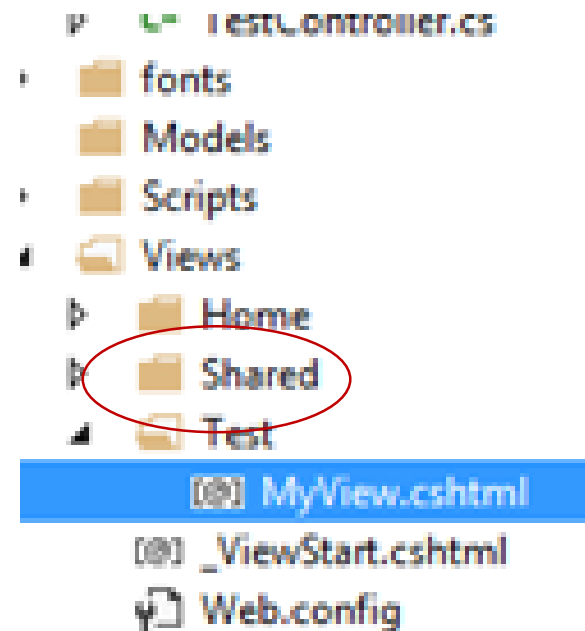
Add Cancel

Controller & View Relation

- Views associated with the particular controller are placed inside a specific folder under views.
 - Folder name needs to match "ControllerName".
 - A controller can access only views located inside its own folder.
- Example: All the views related to Test controller will be placed inside "~ /Views/Test" and Test controller can access only those views which are inside Test folder.

Shared Views

- Views can be reused across multiple controllers.
- Such views will be stored in the “Shared” folder under Views.



Action Method & Views

- An action method can reference more than one view based on the code logic.
- In ASP .NET MVC views and controllers are not tightly coupled:
 - One action method can refer more than one view.
 - One view can be referred by more than one action method (Shared folder)

```
public ActionResult GetView()  
{  
  
    if (Some_Condition_Is_Matching)  
    {  
        return View("MyView");  
    }  
    else  
    {  
        return View("YourView");  
    }  
}
```

Demo: Models

- Add a new model under Models folder
 - Create class Media
 - int ID
 - string Name
- Requirement:
 - Create a page to random action to pick a media and display its details
 - /media/random

Notes:

- To access a model from within a Controller
 - Need to add the namespace.
 - Then you can create objects of that model.
- To access a model from within a View
 - Add a directive "@model" + fully qualified name
 - Within HTML use "@Model.property"

Adding Themes

- ASP .NET application uses Bootstrap as its frontend CSS Framework.
- We can replace with the template we like
 - www.bootswatch.com
 - Download a new one and rename it to: *bootstrap-Name.css*
 - Add it to the project under Content

Update Bootstrap Reference

- App_Start → BundleConfig
 - Combine and bundle web application assets
 - Will reduce number of HTTP requests when a page is loaded → faster page load
- "~/Content/css": contains CSS assets
 - Bootstrap: rename as per your new file → compile
 - Site: generic styles for applications

Demo: Themes

- Download couple of themes.
- Update your app to test them.

ActionResult

- Base class for all action results in ASP .NET MVC.
- Depending on what action is needed
 - An instance of one the derived classes from action results will be returned.
- View method
 - A helper method inherited from the base Controller class.
 - Provide easy creation of a view result.

Action Result Sub Types

Type	Helper Method	Notes
ViewResult	View()	Most popular
PartialViewResult	PartialView()	Return partial view (print function)
ContentResult	Content()	Simple Text
RedirectResult	Redirect()	Go to a URL
RedirectToRouteResult	RedirectToAction()	Redirect to an action
JsonResult	Json()	Return serialized JSON object
FileResult	File()	Example file download
HttpNotFoundResult	HttpNotFound()	Page not found / 404 error
EmptyResult		When an action does need to return any value (void)

Demo: ActionResult

```
return Content("Hello World");
```

```
return HttpNotFound( );
```

```
return new EmptyResult( );
```

```
return RedirectToAction("Index","Home");
```

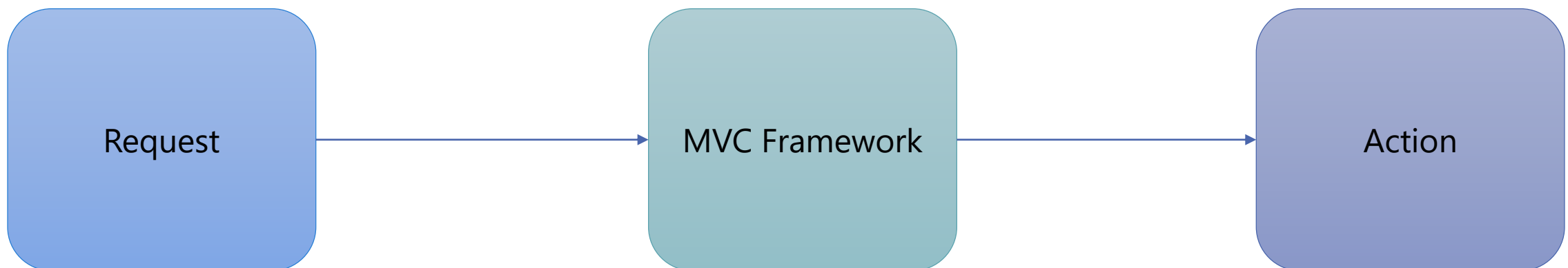
```
return RedirectToAction("Index","Home", new  
{page=1,sortBy="name"});
```

Final Word – Action Results

- No need to know all the details.
 - Know as you use.
 - MSDN is an excellent reference.
- Most used ActionResult
 - View()
 - RedirectToAction()
 - Redirect()
 - HttpNotFound()

Action Parameters

- Action parameters are the inputs to our actions.
- When a request arrives, ASP .NET MVC automatically maps request data to parameter values.
 - If an action requires a parameter, the framework looks for a parameter with the same name in the request data.
 - If parameter with that name exist, the framework will automatically pass their value to the target action.



Forms of Action Parameters

- Embedded in URLs:

/medias/edit/1

- In the query string

/medias/edit?id=1

- In the form data

id=1

Demo: Action with parameter

- Create an action called "edit"
- Pass an integer value in the parameter list called "id"
- The action returns simple content
 - Value of the pass parameter.
- Test
 - Using URL passing
 - Query String

Demo: Action with parameter

- Change the passed integer name to “mediasId”
- Test again
 - Using URL passing
 - Query String
- Check route configuration.

Multiple Parameters

- The default route works for most scenarios.
- There are situations where we need a route with multiple parameters.

Demo: Actions with multiple parameters (1)

A. Create an action method called index.

- Index takes two parameters:
 - int pageIndex
 - string sortBy

B. Lets make the parameters optional

- Provide a default value
pageIndex = 1 , sortBy = name

Demo: Actions with multiple parameters (2)

- Create an action called “released”
- The action requires two parameters
 - Year
 - Month
- Create a custom route to allow the action to run through URL
- Example:
`/media/released/2017/7`

Demo: Actions with multiple parameters (2)

- Add constraints to the route
 - Year is 4 digits, month 2 digits
 - Month has to be between 1 & 12

Q & A

