JOHN ABBOTT
CEGEP/COLLEGE

(420-PS4-AB)
ASP .NET MVC Introduction

Summer 2018

Outline

- ASP. NET Webforms
- What is MVC?
- Development Environment.
- First MVC Web Application.
- Demos: Controller – Views
- Controller & View Relation
- Adding Themes

Webforms VS MVC

ASP .NET Web Forms

- ASP.NET Webforms has served and successfully delivered web application for past years.
  - And still is.

- Reason for success:
  - Rapid application development.
  - Visual programming approach → Visual Studio

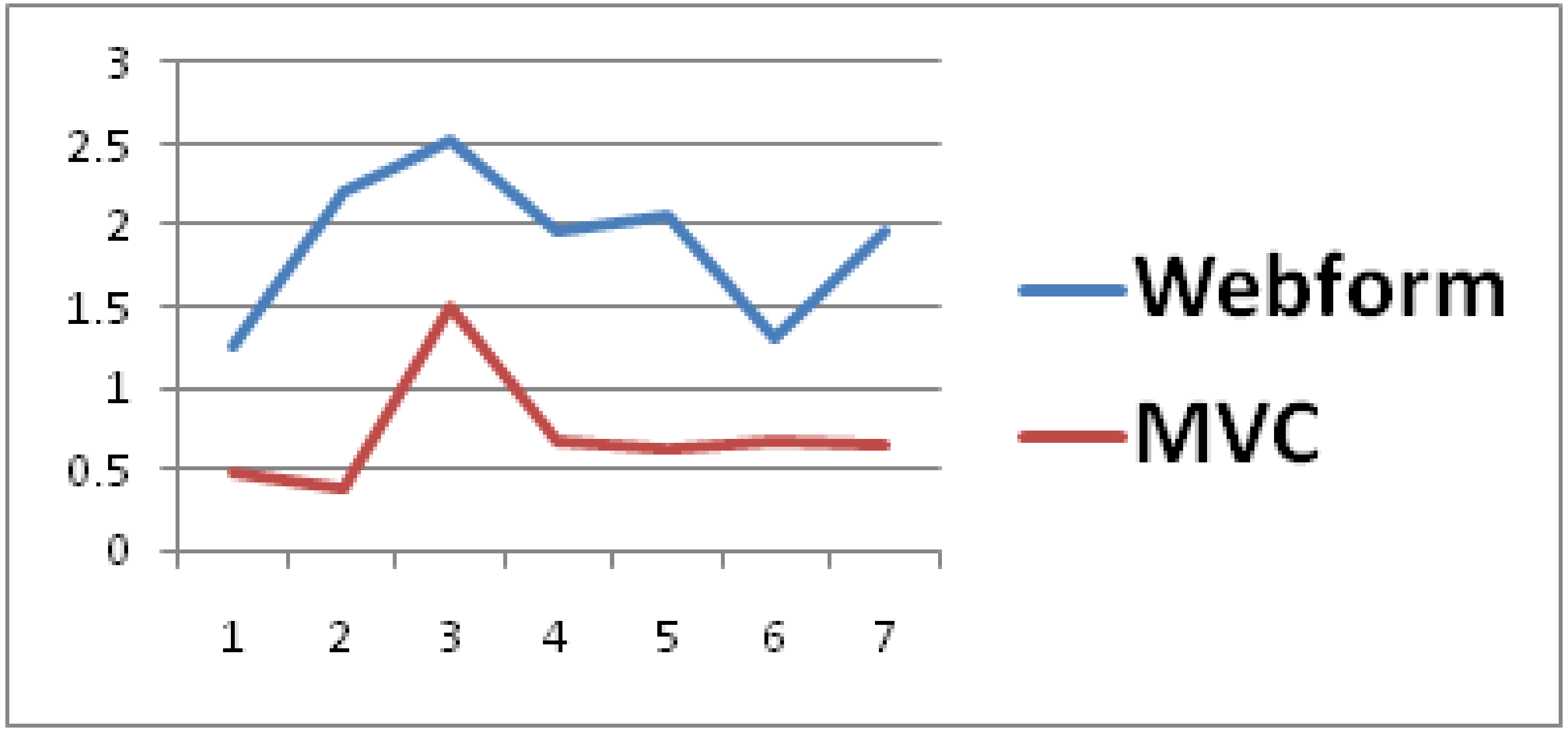
- UI: Drag & drop → backend: code behind

Why create a new technology?

- ASP .NET was a great success.

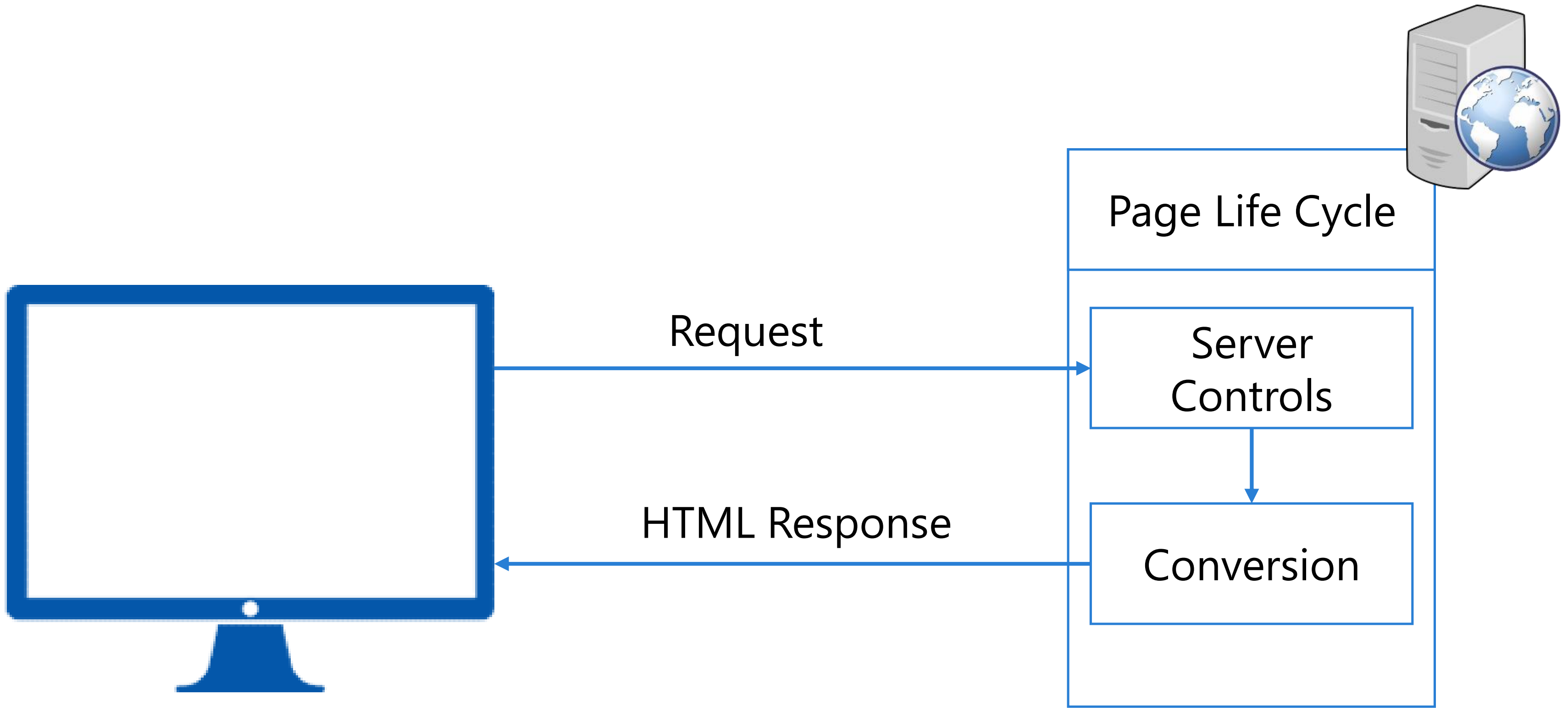
- Bottlenecks:
  - Response time: How fast the server responds to request?.
  - Bandwidth consumption: How much data is generated/sent?

Response Time

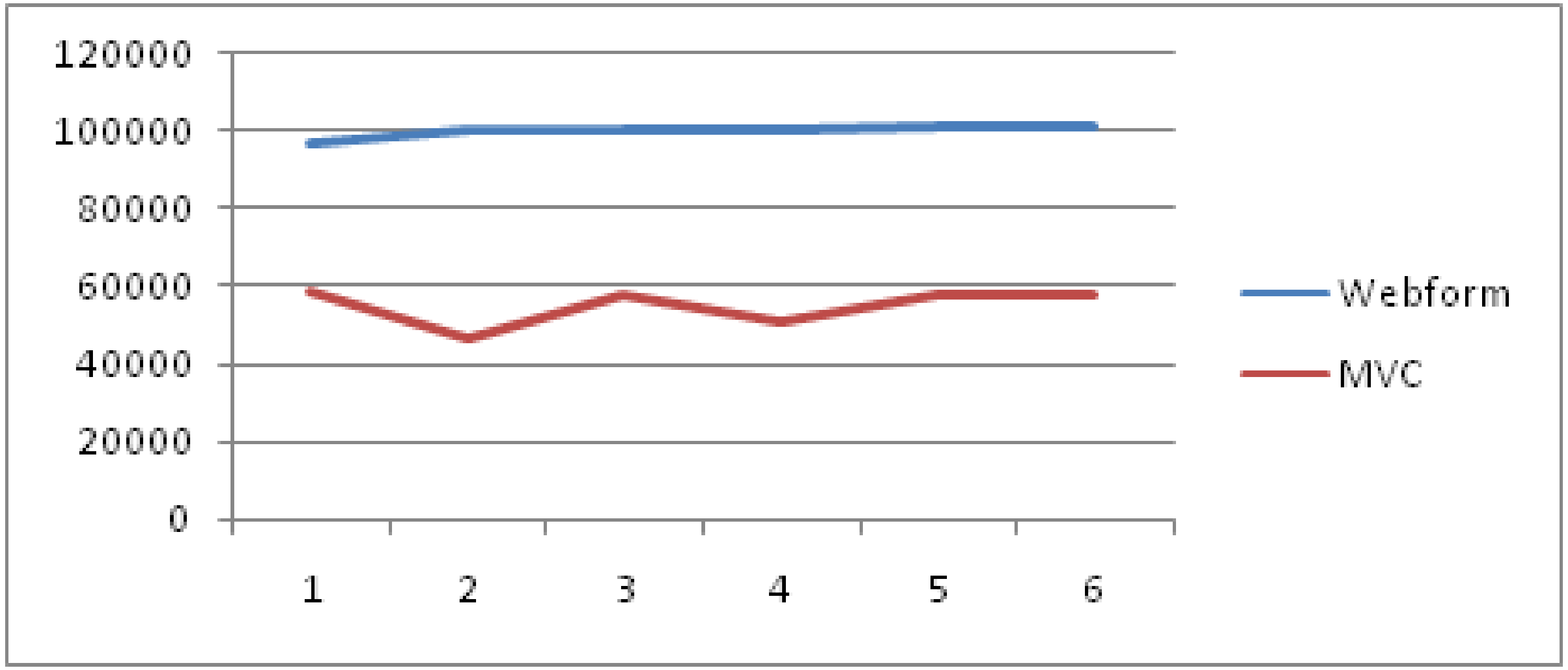


Source: https://www.codeproject.com/Articles/864950/ASP-NET-MVC-vs-ASP-NET-WebForm-performance-compari

APS .NET Web Forms Life Cycle



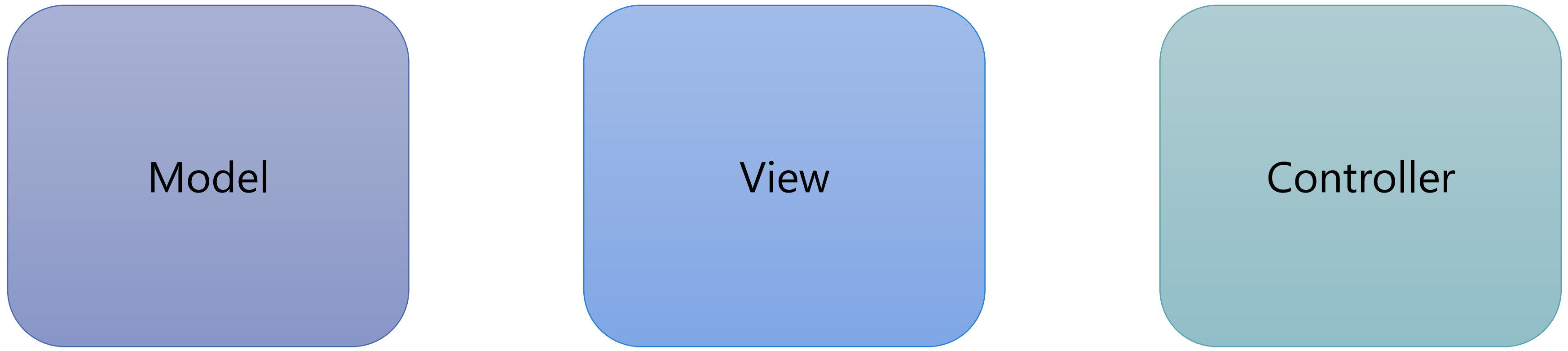
Bandwidth Consumption



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

What is MVC?

MVC Architecture Pattern

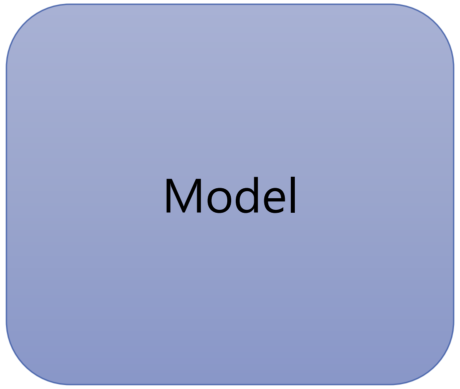


MVC

- 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

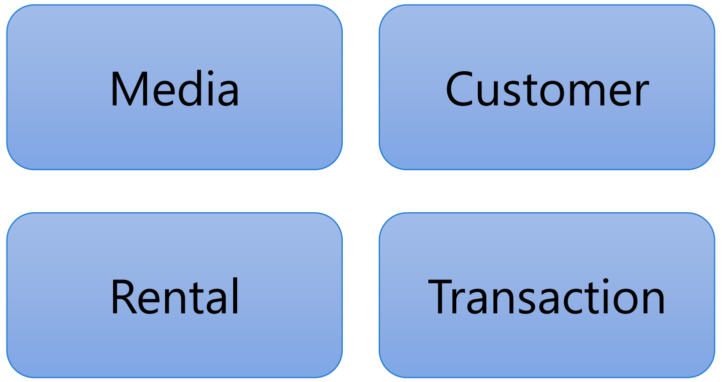
Model

Represents application data and behavior in terms of its problem domain, and independent of UI.



Model Example

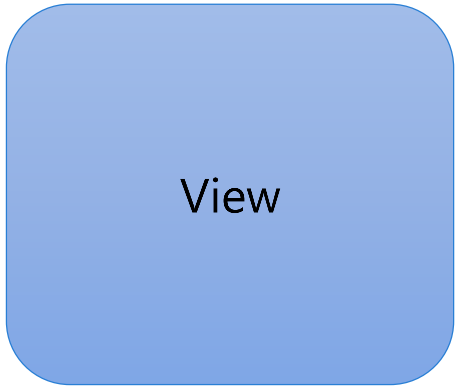
- Video Rental Web Application:
  - Classes



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

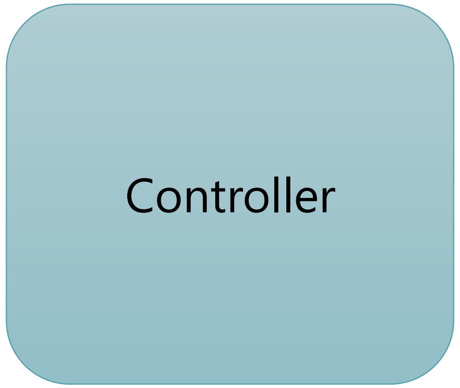
Model

Represents the HTML markup the we display to the user.

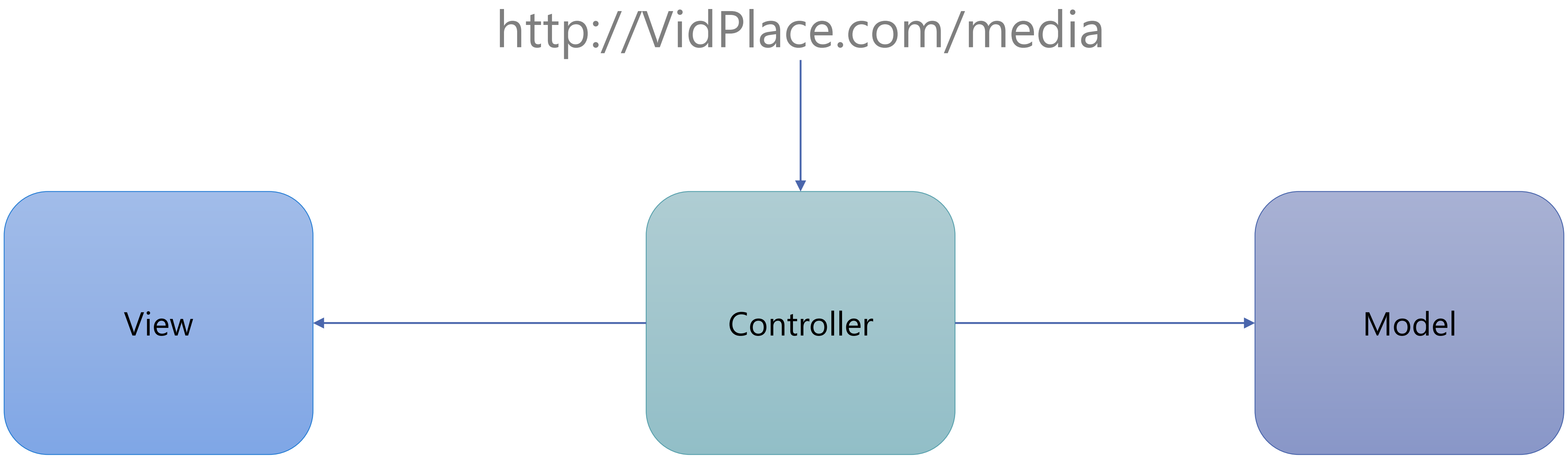


Model

Responsible for handling an HTTP Request.



Example



Better separation of concerns

Better maintainable application

MVC is

IS NOT

- A completely new of doing everything.
- Too complex to learn in time for your project.
- Someone’s attempt to make your life more difficult.

IS

- A useful way to organize code, markup and control flow.
- Easy to learn.
- Designed to help developers build and maintain better applications.

Development Environment

- For MVC 5.0
  - Visual Studio 2013 or later

- Tools → Extensions and Updates → Online
  - Productivity Power Tools
  - Web Essentials

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

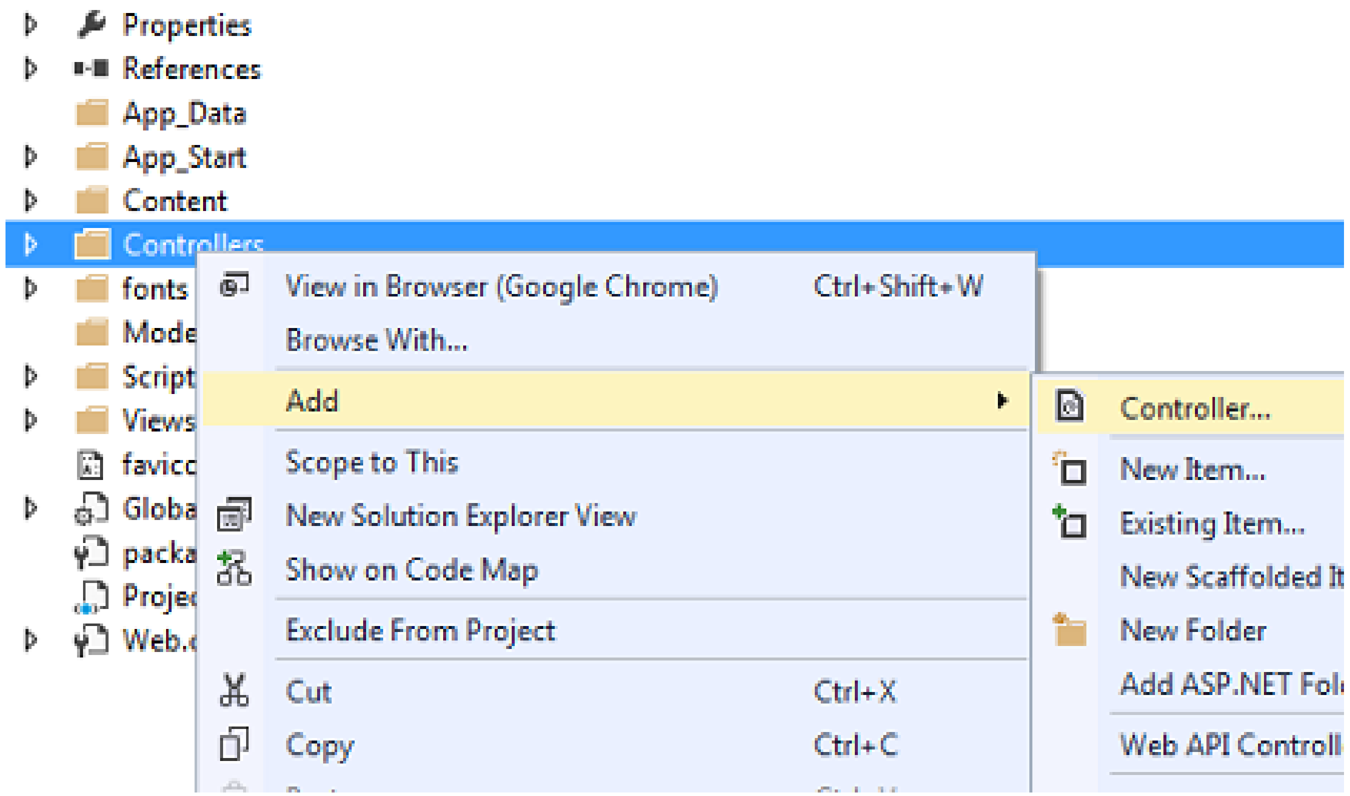
$$\begin{tikzcd} & \text{/media/popular} \arrow[dl] \arrow[dr] & \\ \text{MediaController} & & \text{Popular( )} \end{tikzcd}$$

$$\begin{tikzcd} & \text{/media/edit/1} \arrow[dl] \arrow[dr] & \\ \text{MediaController} & & \text{Edit(int id)} \end{tikzcd}$$

MVC in Action

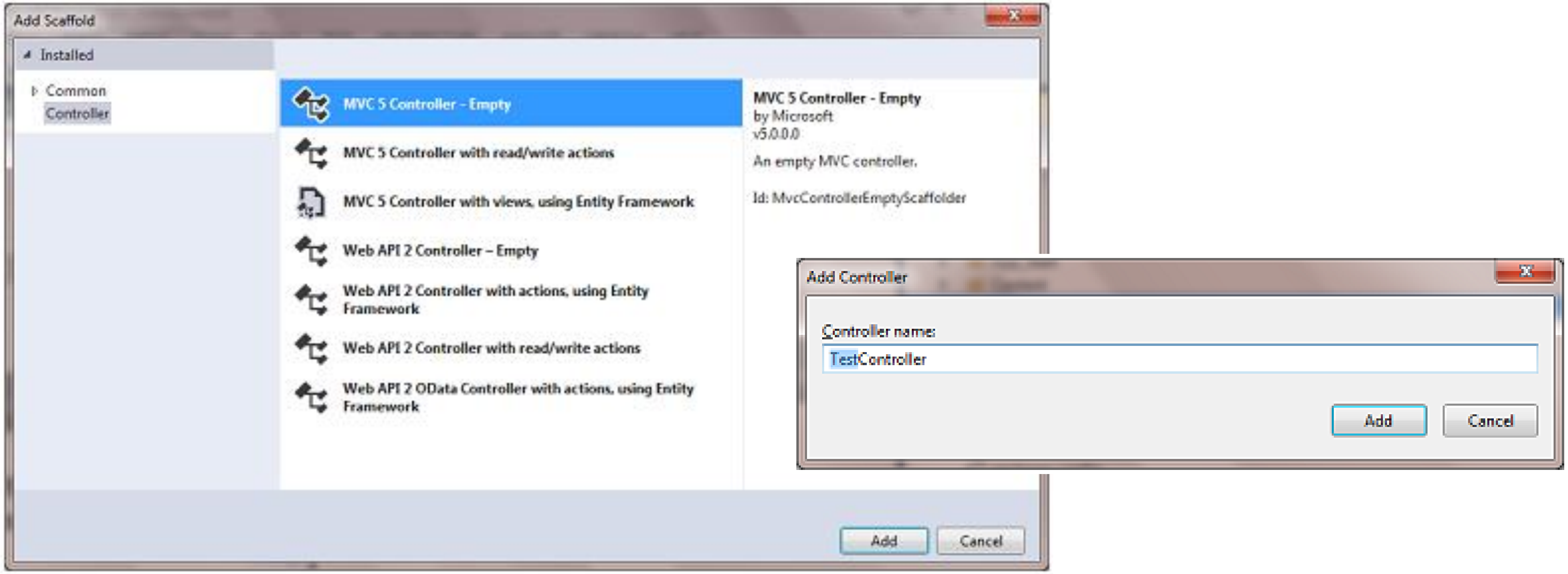
Demo: HelloWorld

- Controllers → Add → Controller



Demo: Snapshot

Select “MVC 5 Controller – Empty” and click Add


Add Scaffold

- Installed
  - Common
  - Controller

MVC 5 Controller – Empty
MVC 5 Controller with read/write actions
MVC 5 Controller with views, using Entity Framework
Web API 2 Controller – Empty
Web API 2 Controller with actions, using Entity Framework
Web API 2 Controller with read/write actions
Web API 2 OData Controller with actions, using Entity Framework

MVC 5 Controller - Empty
by Microsoft
v5.0.0.0
An empty MVC controller.
Id: MvcControllerEmptyScaffolder

Add Cancel

Add Controller

Controller name: TestController

Add Cancel

Demo: Controller (1)

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

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

Demo: Views

- Under the TestController, add

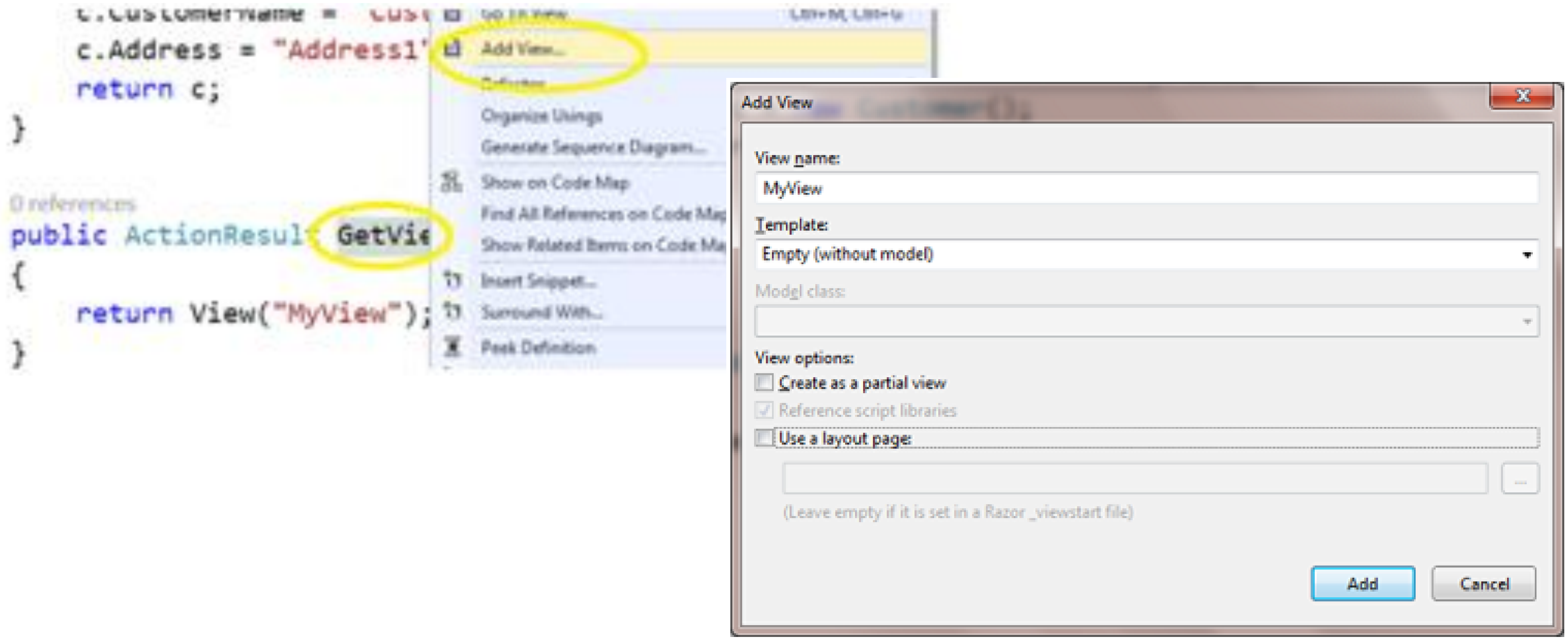
```
public ActionResult GetView()
{
    return View("MyView");
}
```

Demo: Controller – Returning Objects

```
public class Customer
{
  public string Id { get; set; }
  public string Name { get; set; }
}
public class TestController : Controller
{
  public Customer GetCustomer()
  {
    Customer c = new Customer();
    c.Id = "1001";
    c.Name = "John Abbott";
    return c;
  }
}
```

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

Demo: Add View



```
        c.CustomerName = Cust
        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)

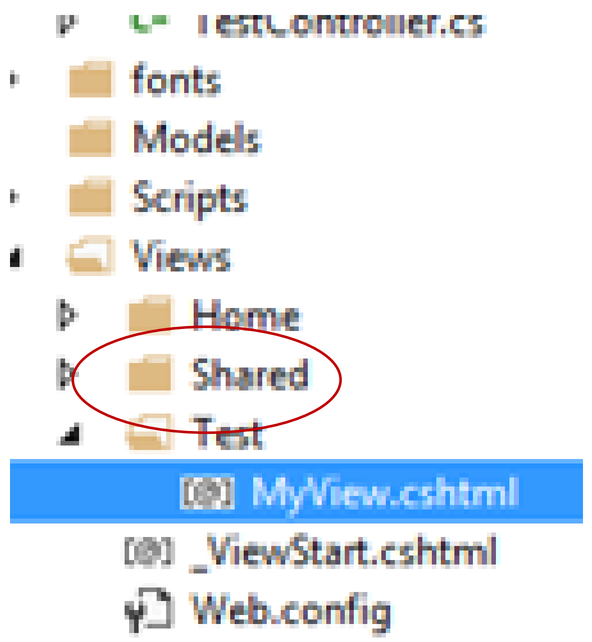
Controller & View Relation

- 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: Using Models

- Add a new model under Models folder
  - Create class Media
    - int ID
    - string Name
- Requirement:
  - Create a page to randomly pick a media and show 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”

Themes

Adding Themes

- ASP .NET application uses Bootstrap as its frontend CSS Framework.

- We can replace with the template we like
  - www.bootswatch.com/3/
  - 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

Exercise: Themes

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

Q & A

JOHN ABBOTT
CEGEP/COLLEGE