

MVC_Moving_Data_201X

Add the different ways of creating content to your pages/views.... Test “hard coded” then “Separate the Concerns” by moving any <script> or <style> to its own file – or from the View to a Controller Action Result.

Do this by either adding a new controller and add methods to your different views – or just methods to an existing view.

- **Remember** a new controller adds a new URL entry
 - testController.cs would be [http://localhost:\(port\)/test](http://localhost:(port)/test)
 - Each method in testController.cs would map to a view maybe like:
 - JSONView
 - XMLView
 - SQLView

From a source to a destination: JSON, XML & The Querystring.... Not to be left out: The VIEWBAG..

XMLHttpRequest from file:

- Hard coded in View (or HTML page)
 - JavaScript in <body>: https://www.w3schools.com/js/tryit.asp?filename=tryjs_where_to_body
- View joined to .js file with <script src =>
 - External JavaScript: https://www.w3schools.com/js/tryit.asp?filename=tryjs_where_to_external

Added as a result in a controller ActionResult (JsonResult shown in examples)

- ViewResult - Renders a specified view to the response stream
 - PartialViewResult - Renders a specified partial view to the response stream
 - EmptyResult - An empty response is returned
 - RedirectResult - Performs an HTTP redirection to a specified URL
 - RedirectToRouteResult - Performs an HTTP redirection to a URL that is determined by the routing engine, based on given route data
 - JsonResult - Serializes a given ViewData object to JSON format
 - JavaScriptResult - Returns a piece of JavaScript code that can be executed on the client
 - ContentResult - Writes content to the response stream without requiring a view
 - FileContentResult - Returns a file to the client
 - FileStreamResult - Returns a file to the client, which is provided by a Stream
 - FilePathResult - Returns a file to the client
- HelloWorld: Using the querystring and route data parameters – from MvcMovie.

Begin Examples

Begin ViewBag

Example: ViewBag -How to create two properties for the ViewBag in a controller in C#.

```
public ActionResult Index()
{
    ViewBag.MyMessageToUsers = "Hello from me.";
    ViewBag.AnswerText = "Your answer goes here.";

    return View();
}
```

How to retrieve those ViewBag properties from your view. The first line displays the value of the MyMessageToUsers property as text. The second line creates a text box and pre-populates it with the value of the AnswerText property.

```
@ViewBag.MyMessageToUsers
@Html.TextBox("AnswerText")
```

End ViewBag

These examples require the external file mapped in the code. Suggest creating a folder for "Support" files – xhttp.open maps to - support/ [xmlhttp_info.txt](#)

Example: AJAX: Update parts of a web page, without reloading the whole page:

https://www.w3schools.com/xml/tryit.asp?filename=tryxml_httprequest

```
<body>

<h2>Using the XMLHttpRequest Object</h2>

<div id="demo">
<button type="button" onclick="loadXMLDoc()">Change Content</button>
</div>

<script>
function loadXMLDoc() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            document.getElementById("demo").innerHTML =
                this.responseText;
        }
    };
    xhttp.open("GET", "xmlhttp_info.txt", true);
    xhttp.send();
}
</script>

</body>
```

Content of `xmlhttp_info.txt`

```
<h1>AJAX</h1>
<p>AJAX is not a programming language.</p>
<p>AJAX is a technique for accessing web servers from a web page.</p>
<p>AJAX stands for Asynchronous JavaScript And XML.</p>
<p>With the XMLHttpRequest object you can update parts of a web page, without reloading the whole page.</p>
<p>The XMLHttpRequest object is used to exchange data with a server behind the scenes.</p>
```

End AJAX

Example: JSON Http Request: https://www.w3schools.com/js/tryit.asp?filename=tryjson_http

```
<body>

<div id="id01"></div>

<script>
var xmlhttp = new XMLHttpRequest();
var url = "myTutorials.txt";

xmlhttp.onreadystatechange = function() {
  if (this.readyState == 4 && this.status == 200) {
    var myArr = JSON.parse(this.responseText);
    myFunction(myArr);
  }
};
xmlhttp.open("GET", url, true);
xmlhttp.send();

function myFunction(arr) {
  var out = "";
  var i;
  for(i = 0; i < arr.length; i++) {
    out += '<a href="' + arr[i].url + '"' +
    arr[i].display + '</a><br>';
  }
  document.getElementById("id01").innerHTML = out;
}
</script>

</body>
```

Content of [myTutorials.txt](#) – A good way to populate navigation – style with CSS:
https://www.w3schools.com/css/tryit.asp?filename=trycss_dropdown_navbar

```
[
{
"display": "HTML Tutorial",
"url": "https://www.w3schools.com/html/default.asp"
},
{
"display": "CSS Tutorial",
"url": "https://www.w3schools.com/css/default.asp"
},
{
"display": "JavaScript Tutorial",
"url": "https://www.w3schools.com/js/default.asp"
},
{
"display": "jQuery Tutorial",
"url": "https://www.w3schools.com/jquery/default.asp"
},
{
"display": "SQL Tutorial",
"url": "https://www.w3schools.com/sql/default.asp"
},
{
"display": "PHP Tutorial",
"url": "https://www.w3schools.com/php/default.asp"
},
{
"display": "XML Tutorial",
"url": "https://www.w3schools.com/xml/default.asp"
}
]
```

End JSON

Example: Retrieve data from an XML file and display the data in an HTML table:

https://www.w3schools.com/xml/tryit.asp?filename=try_dom_xmlhttprequest_xml

```
<body>

<p><button onclick="loadXMLDoc()">Get CD info</button></p>

<table id="demo" border="1">
<tr><th>Artist</th><th>Title</th></tr>
</table>

<script>
function loadXMLDoc() {
  var xmlhttp = new XMLHttpRequest();
  xmlhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      myFunction(this);
    }
  };
  xmlhttp.open("GET", "cd_catalog.xml", true);
  xmlhttp.send();
}

function myFunction(xml) {
  var x, i, xmlDoc, table;
  xmlDoc = xml.responseXML;
  table = "<tr><th>Artist</th><th>Title</th></tr>";
  x = xmlDoc.getElementsByTagName("CD")
  for (i = 0; i < x.length; i++) {
    table += "<tr><td>" +
      x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
      "</td><td>" +
      x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
      "</td></tr>";
  }
  document.getElementById("demo").innerHTML = table;
}
</script>
```

Contents of [cd_catalog.xml](https://www.w3schools.com/xml/cd_catalog.xml): https://www.w3schools.com/xml/cd_catalog.xml

```
<CATALOG>
<CD>
<TITLE>Empire Burlesque</TITLE>
<ARTIST>Bob Dylan</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>Columbia</COMPANY>
<PRICE>10.90</PRICE>
<YEAR>1985</YEAR>
</CD>
<CD>
<TITLE>Hide your heart</TITLE>
```

<ARTIST>Bonnie Tyler</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>CBS Records</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1988</YEAR>
</CD>
<CD>
<TITLE>Greatest Hits</TITLE>
<ARTIST>Dolly Parton</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>RCA</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1982</YEAR>
</CD>
<CD>
<TITLE>Still got the blues</TITLE>
<ARTIST>Gary Moore</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Virgin records</COMPANY>
<PRICE>10.20</PRICE>
<YEAR>1990</YEAR>
</CD>
<CD>
<TITLE>Eros</TITLE>
<ARTIST>Eros Ramazzotti</ARTIST>
<COUNTRY>EU</COUNTRY>
<COMPANY>BMG</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1997</YEAR>
</CD>
<CD>
<TITLE>One night only</TITLE>
<ARTIST>Bee Gees</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Polydor</COMPANY>
<PRICE>10.90</PRICE>
<YEAR>1998</YEAR>
</CD>
<CD>
<TITLE>Sylvias Mother</TITLE>
<ARTIST>Dr.Hook</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>CBS</COMPANY>
<PRICE>8.10</PRICE>
<YEAR>1973</YEAR>
</CD>
<CD>
<TITLE>Maggie May</TITLE>
<ARTIST>Rod Stewart</ARTIST>

```
<COUNTRY>UK</COUNTRY>
<COMPANY>Pickwick</COMPANY>
<PRICE>8.50</PRICE>
<YEAR>1990</YEAR>
</CD>
<CD>
<TITLE>Romanza</TITLE>
<ARTIST>Andrea Bocelli</ARTIST>
<COUNTRY>EU</COUNTRY>
<COMPANY>Polydor</COMPANY>
<PRICE>10.80</PRICE>
<YEAR>1996</YEAR>
</CD>
<CD>
<TITLE>When a man loves a woman</TITLE>
<ARTIST>Percy Sledge</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>Atlantic</COMPANY>
<PRICE>8.70</PRICE>
<YEAR>1987</YEAR>
</CD>
<CD>
<TITLE>Black angel</TITLE>
<ARTIST>Savage Rose</ARTIST>
<COUNTRY>EU</COUNTRY>
<COMPANY>Mega</COMPANY>
<PRICE>10.90</PRICE>
<YEAR>1995</YEAR>
</CD>
<CD>
<TITLE>1999 Grammy Nominees</TITLE>
<ARTIST>Many</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>Grammy</COMPANY>
<PRICE>10.20</PRICE>
<YEAR>1999</YEAR>
</CD>
<CD>
<TITLE>For the good times</TITLE>
<ARTIST>Kenny Rogers</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Mucik Master</COMPANY>
<PRICE>8.70</PRICE>
<YEAR>1995</YEAR>
</CD>
<CD>
<TITLE>Big Willie style</TITLE>
<ARTIST>Will Smith</ARTIST>
<COUNTRY>USA</COUNTRY>
```

<COMPANY>Columbia</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1997</YEAR>
</CD>
<CD>
<TITLE>Tupelo Honey</TITLE>
<ARTIST>Van Morrison</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Polydor</COMPANY>
<PRICE>8.20</PRICE>
<YEAR>1971</YEAR>
</CD>
<CD>
<TITLE>Soulsville</TITLE>
<ARTIST>Jorn Hoel</ARTIST>
<COUNTRY>Norway</COUNTRY>
<COMPANY>WEA</COMPANY>
<PRICE>7.90</PRICE>
<YEAR>1996</YEAR>
</CD>
<CD>
<TITLE>The very best of</TITLE>
<ARTIST>Cat Stevens</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Island</COMPANY>
<PRICE>8.90</PRICE>
<YEAR>1990</YEAR>
</CD>
<CD>
<TITLE>Stop</TITLE>
<ARTIST>Sam Brown</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>A and M</COMPANY>
<PRICE>8.90</PRICE>
<YEAR>1988</YEAR>
</CD>
<CD>
<TITLE>Bridge of Spies</TITLE>
<ARTIST>T'Pau</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Siren</COMPANY>
<PRICE>7.90</PRICE>
<YEAR>1987</YEAR>
</CD>
<CD>
<TITLE>Private Dancer</TITLE>
<ARTIST>Tina Turner</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Capitol</COMPANY>

<PRICE>8.90</PRICE>
<YEAR>1983</YEAR>
</CD>
<CD>
<TITLE>Midt om natten</TITLE>
<ARTIST>Kim Larsen</ARTIST>
<COUNTRY>EU</COUNTRY>
<COMPANY>Medley</COMPANY>
<PRICE>7.80</PRICE>
<YEAR>1983</YEAR>
</CD>
<CD>
<TITLE>Pavarotti Gala Concert</TITLE>
<ARTIST>Luciano Pavarotti</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>DECCA</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1991</YEAR>
</CD>
<CD>
<TITLE>The dock of the bay</TITLE>
<ARTIST>Otis Redding</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>Stax Records</COMPANY>
<PRICE>7.90</PRICE>
<YEAR>1968</YEAR>
</CD>
<CD>
<TITLE>Picture book</TITLE>
<ARTIST>Simply Red</ARTIST>
<COUNTRY>EU</COUNTRY>
<COMPANY>Elektra</COMPANY>
<PRICE>7.20</PRICE>
<YEAR>1985</YEAR>
</CD>
<CD>
<TITLE>Red</TITLE>
<ARTIST>The Communards</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>London</COMPANY>
<PRICE>7.80</PRICE>
<YEAR>1987</YEAR>
</CD>
<CD>
<TITLE>Unchain my heart</TITLE>
<ARTIST>Joe Cocker</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>EMI</COMPANY>
<PRICE>8.20</PRICE>

```
<YEAR>1987</YEAR>
</CD>
</CATALOG>
```

End XML

Example: Read Customer data from SQL – Format as JSON – Style with CSS: Must be live on the web to work:https://www.w3schools.com/js/tryit.asp?filename=tryjson_server_sql_style

```
<body>
<style>
h1 {
  border-bottom: 3px solid #cc9900;
  color: #996600;
  font-size: 30px;
}
table, th, td {
  border: 1px solid grey;
  border-collapse: collapse;
  padding: 5px;
}
table tr:nth-child(odd) {
  background-color: #f1f1f1;
}
table tr:nth-child(even) {
  background-color: #ffffff;
}
</style>

<h2>Customers</h2>
<div id="id01"></div>

<script>
var xmlhttp = new XMLHttpRequest();
var url = "https://www.w3schools.com/js/Customers_MYSQL.php";

xmlhttp.onreadystatechange=function() {
  if (this.readyState == 4 && this.status == 200) {
    myFunction(this.responseText);
  }
}
xmlhttp.open("GET", url, true);
xmlhttp.send();

function myFunction(response) {
  var arr = JSON.parse(response);
  var i;
  var out = "<table>";

  for(i = 0; i < arr.length; i++) {
    out += "<tr><td>" +
```

```

        arr[i].Name +
        "</td><td>" +
        arr[i].City +
        "</td><td>" +
        arr[i].Country +
        "</td></tr>";
    }
    out += "</table>";
    document.getElementById("id01").innerHTML = out;
}
</script>

</body>

```

Form Handler: https://www.w3schools.com/js/Customers_MYSQL.php - Note SELECT and \$outp statements

```

<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");

$conn = new mysqli("myServer", "myUser", "myPassword", "Northwind");

$result = $conn->query("SELECT CompanyName, City, Country FROM Customers");

$outp = "[";
while($rs = $result->fetch_array(MYSQLI_ASSOC)) {
    if ($outp != "[") {$outp .= ",";}
    $outp .= '{"Name":"' . $rs["CompanyName"] . "',';
    $outp .= '"City":"' . $rs["City"] . "',';
    $outp .= '"Country":"' . $rs["Country"] . "'}';
}
$outp .= "]";

$conn->close();

echo($outp);
?>

```

End SQL

Example: HelloWorld Part 1: Routing & Querystring (from MvcMovie): <https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/adding-a-controller>

- Passing information from the controller to a view; the whole reason for this exercise...

1. In Solution Explorer, right-click the Controllers folder and then click Add, then Controller.
2. In the Add Scaffold dialog box, click MVC 5 Controller - Empty, and then click Add.
3. Name your new controller "HelloWorldController" and click Add.
 - a. Notice in Solution Explorer that a new file has been created named HelloWorldController.cs and a new folder Views\HelloWorld
4. Replace the contents of the file with the following code.

```
using System.Web;
using System.Web.Mvc;

namespace MvcMovie.Controllers
{
    public class HelloWorldController : Controller
    {
        //
        // GET: /HelloWorld/

        public string Index()
        {
            return "This is my <b>default</b> action...";
        }

        //
        // GET: /HelloWorld/Welcome/

        public string Welcome()
        {
            return "This is the Welcome action method...";
        }
    }
}
```

5. Run the application (press F5 or Ctrl+F5). In the browser, append "HelloWorld" to the path in the address bar. (For example: <http://localhost:1234/HelloWorld>.)
6. You set the format for routing in the App_Start/RouteConfig.cs file.

```
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 }
    );
}
```

7. Browse to <http://localhost:xxxx/HelloWorld/Welcome>. The Welcome method runs and returns the string "This is the Welcome action method..."

8. Change your Welcome method in HelloWorld controller to include two parameters; this can pass some parameter information from the URL to the controller (for example, /HelloWorld/Welcome?name=Scott&numtimes=4)

```
public string Welcome(string name, int numTimes = 1) {
    return HttpUtility.HtmlEncode("Hello " + name + ", NumTimes is: " + numTimes);
}
```

9. Browse to URL (<http://localhost:xxxx/HelloWorld/Welcome?name=Scott&numtimes=4>). Try different values for name and numtimes in the URL.

10. **name and numTimes parameters are passed as query strings.** The ? (question mark) in the above URL is a separator, and the query strings follow. The & character separates query strings.

11. Replace the Welcome method with the following code:

```
public string Welcome(string name, int ID = 1)
{
    return HttpUtility.HtmlEncode("Hello " + name + ", ID: " + ID);
}
```

12. Run the application - enter the following URL: <http://localhost:xxx/HelloWorld/Welcome/3?name=Rick>
a. The third URL segment matched the route parameter ID. The Welcome action method contains a parameter (ID) that matched the URL specification in the RegisterRoutes method.

13. Add a route to pass both the name and numtimes in parameters as route data in the URL.
a. In the App_Start\RouteConfig.cs file, add the "Hello" route:

```
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 }
        );

        routes.MapRoute(
            name: "Hello",
            url: "{controller}/{action}/{name}/{id}"
        );
    }
}
```

14. Browse to /localhost:XXX/HelloWorld/Welcome/Scott/3

HelloWorld Part 2: Add a View

15. modify the HelloWorldController class to use view by changing the Index method to return a View

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

16. Right click the Views\HelloWorld folder and click Add, then click MVC 5 View Page with (Layout Razor).
a. In the Specify Name for Item dialog box, enter Index, and then click OK.
b. In the Select a Layout Page dialog, accept the default _Layout.cshtml and click OK.

c. The MvcMovie\Views\HelloWorld\Index.cshtml file is created; Add the following:

```
@{
    Layout = "~/Views/Shared/_Layout.cshtml";
}

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

<p>Hello from our View Template!</p>
```

17. Browse to the HelloWorld controller <http://localhost:xxxx/HelloWorld> - note results

18. **SKIPPED** Modifying the _layout.cshtml – title, links etc... instructions... can you do it anyway?

Passing information from the controller to a view; the whole reason for this exercise...

Best practice: A view template should never perform business logic or interact with a database directly.

19. Return to the HelloWorldController.cs file and change the Welcome method to add a Message and NumTimes value to the ViewBag.

a. The complete HelloWorldController.cs file looks like this:

```
using System.Web;
using System.Web.Mvc;

namespace MvcMovie.Controllers
{
    public class HelloWorldController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult Welcome(string name, int numTimes = 1)
        {
            ViewBag.Message = "Hello " + name;
            ViewBag.NumTimes = numTimes;

            return View();
        }
    }
}
```

20. In the Build menu, **select Build Solution** (or Ctrl+Shift+B) to make sure the project is compiled.

- Right click the Views\HelloWorld folder and click Add, then click MVC 5 View Page with Layout (Razor).
- In the Specify Name for Item dialog box, enter Welcome, and then click OK.
- In the Select a Layout Page dialog, accept the default _Layout.cshtml and click OK.
- The MvcMovie\Views\HelloWorld\Welcome.cshtml file is created.

21. Replace the markup in the Welcome.cshtml file.

- Create a loop that says "Hello" as many times as the user says it should.

b. The complete Welcome.cshtml file:

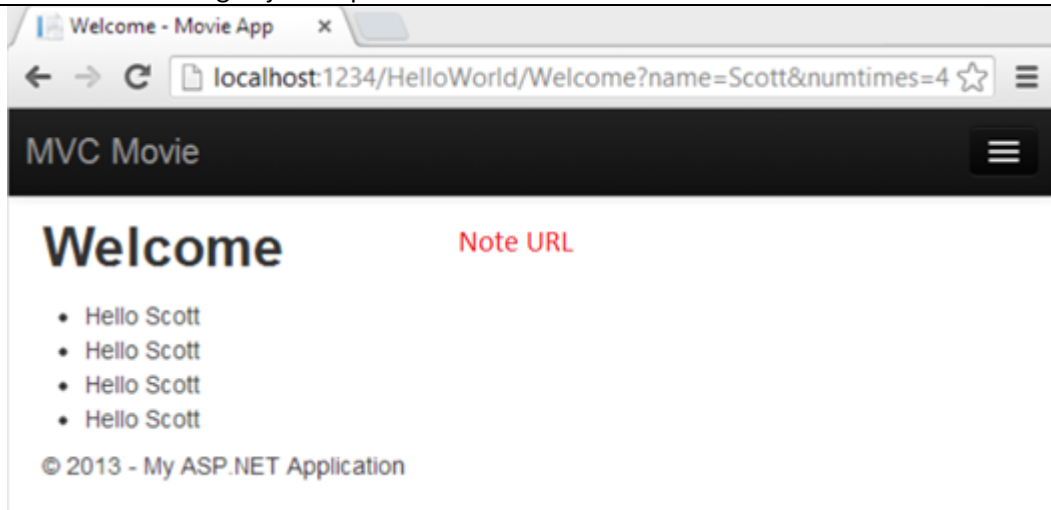
```
@{
    ViewBag.Title = "Welcome";
}

<h2>Welcome</h2>

<ul>
    @for (int i = 0; i < ViewBag.NumTimes; i++)
    {
        <li>@ViewBag.Message</li>
    }
</ul>
```

22. Browse to: <http://localhost:xx/HelloWorld/Welcome?name=Scott&numtimes=4>

a. A ViewBag object to pass data from the controller to a view



End HelloWorld & the Querystring

Begin JsonResult

Example 1: How to return an instance of the JsonResult class from an action method:[https://msdn.microsoft.com/en-us/library/system.web.mvc.jsonresult\(v=vs.118\).aspx](https://msdn.microsoft.com/en-us/library/system.web.mvc.jsonresult(v=vs.118).aspx)

```
public ActionResult Movies()
{
    var movies = new List<object>();

    movies.Add(new { Title = "Ghostbusters", Genre = "Comedy", Year = 1984 });
    movies.Add(new { Title = "Gone with Wind", Genre = "Drama", Year = 1939 });
    movies.Add(new { Title = "Star Wars", Genre = "Science Fiction", Year = 1977 });

    return Json(movies, JsonRequestBehavior.AllowGet);
}
```

How to retrieve and display the JSON-formatted content.

```
<input name="btnGetMovies" id="btnGetMovies" type="submit" value="Get Movies">
<ul id="movieList"></ul>

<script src="~/Scripts/jquery-1.10.2.js"></script>
<script type="text/javascript">
    $("#btnGetMovies").click(function () {
        var actionUrl = '@Url.Action("Movies", "Home")';
        $.getJSON(actionUrl, displayData);
    });

    function displayData(response) {
        if (response != null) {
            for (var i = 0; i < response.length; i++) {
                $("#movieList").append("<li>" + response[i].Title + " " + response[i].Genre + " " + response[i].Year +
"</li>")
            }
        }
    }
</script>
```

End JsonResult – 1

Begin JsonResult - 2

Example: JsonResult – 2: Add to (Home) Controller

```
//Json Result

public JsonResult ActionDemo()

{

    UserData obj = new UserData() { UserName = "Test Name", Email = "test@test.com",
    Phone = "34234234" };

    return Json(obj, , JsonRequestBehavior.AllowGet);

}

public class UserData

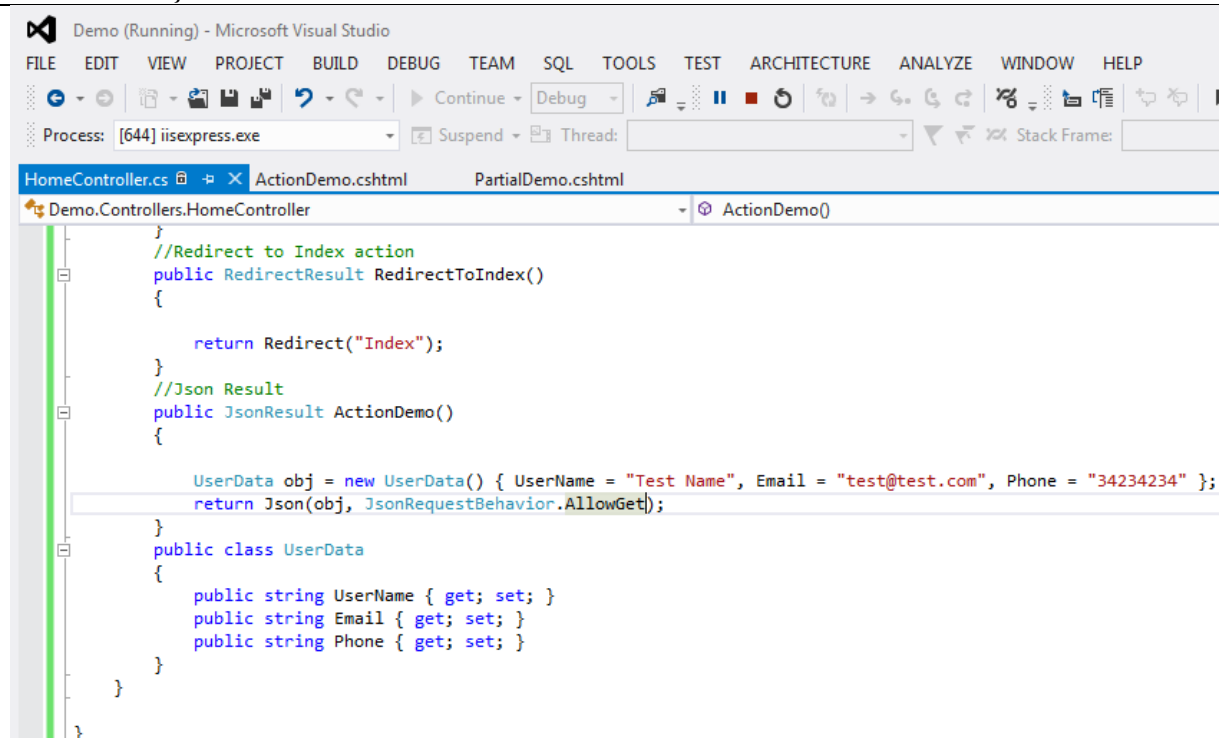
{

    public string UserName { get; set; }

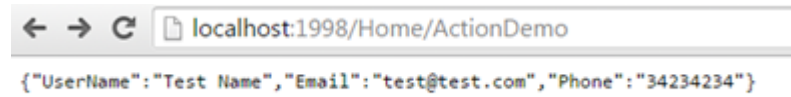
    public string Email { get; set; }

    public string Phone { get; set; }

}
```



Output not associated with a styled view....



```
{ "UserName": "Test Name", "Email": "test@test.com", "Phone": "34234234" }
```

End JsonResult - 2

Resources

1. MvcMovie: Using Routes & Querystring
 - a. Adding a Controller: <https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/adding-a-controller>
 - b. Adding a View: <https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/adding-a-view>
2. Build an ASP.NET app in Azure with SQL Database: <https://docs.microsoft.com/en-us/azure/app-service/app-service-web-tutorial-dotnet-sqldatabase>
3. JsonResult Type in MVC: <http://www.c-sharpcorner.com/UploadFile/2ed7ae/jsonresult-type-in-mvc/>