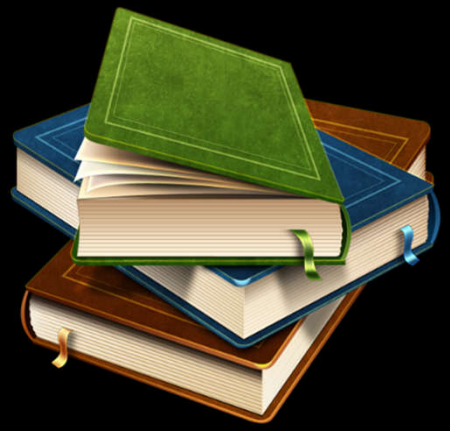
AJAX in ASP.NET MVC

AJAX, Partial Page Rendering, jQuery AJAX, MVC AJAX Helpers

Table of Contents

- What is AJAX?
  - Raw AJAX vs. AJAX using a JS Library
  - Partial Page Rendering vs. JSON Service
- AJAX with Unobtrusive JavaScript
- AJAX MVC Helpers
  - `ActionLink` and `BeginForm`
  - Partial Views and AJAX
- JSON, AJAX and ASP.NET MVC



What is AJAX?

Asynchronous JavaScript and XML

AJAX

- AJAX is acronym of Asynchronous JavaScript and XML
  - AJAX == technique for asynchronously loading (in the background) of dynamic Web content and data from the Web server into a HTML page
  - Allows dynamically changing the DOM (client-side) in Web applications
- Two styles of AJAX
  - Partial page rendering
    - Load an HTML fragment and display it in a `<div>`
  - JSON service with client-side rendering
    - Loading a JSON object and render it at the client-side with

- Examples of applications using AJAX: Gmail, Google Maps, Youtube, and Facebook tabs.

AJAX: Pros and Cons

- AJAX advantages
  - Asynchronous calls → data is loaded after the page is shown
  - Minimal data transfer → less traffic, fast update
  - Responsiveness → better user experience
- AJAX disadvantages
  - Requires more development efforts
  - The browser [Back] and [Refresh] buttons don't work
  - Might cause SEO problems: search engine bots may skip the AJAX

The XMLHttpRequest Object

- Browsers support **`XMLHttpRequest`** object
  - Exposes a JavaScript API to send raw AJAX requests
  - Send HTTP / HTTPS requests directly to the Web server
    - GET / POST / PUT / DELETE / other
  - AJAX response might be JSON, XML, HTML, as plain text, etc.
- Same-origin policy
  - https://en.wikipedia.org/wiki/Same-origin_policy
  - AJAX requests can only access the same server that served the original Web page executing the AJAX call

Raw AJAX – Example

```
function updateServerTimeAjax() {
  var xhr = new XMLHttpRequest();
  xhr.open("GET", "/Home/ServerTime", true);
  xhr.onreadystatechange = function() {
    var timeDiv = document.getElementById("timeDisplay");
    timeDiv.innerHTML = '';
    if (xhr.readyState == 4 && xhr.status == 200) {
      timeDiv.innerHTML = xhr.responseText;
    }
  }
  xhr.send();
}
```

The request is finished and the response is ready

HTTP status code is "200 OK"

AJAX with jQuery – Example

- jQuery dramatically simplifies working with AJAX:

```
<div id="timeDisplay"></div>

<a onclick="updateServerTimeAjax ()">Get Server Time</a>

<script>
  function updateServerTimeAjax () {
    $("#timeDisplay").load("/Home/ServerTime");
  }
</script>
```

jQuery AJAX Methods

Method	Description
$.ajax()	Performs an async AJAX request
$.ajaxSetup()	Sets the default values for future AJAX requests
$.get()	Loads data from a server using an AJAX HTTP GET request
$.getJSON()	Loads JSON-encoded data from a server using a HTTP GET request
$.getScript()	Loads (and executes) a JavaScript from a server using an AJAX HTTP GET request
$.post()	Loads data from a server using an AJAX HTTP POST request
ajaxComplete()	Specifies a function to run when the AJAX request completes

jQuery AJAX Methods

Method	Description
ajaxError()	Specifies a function to run when the AJAX request completes with an error
ajaxSend()	Specifies a function to run before the AJAX request is sent
ajaxStart()	Specifies a function to run when the first AJAX request begins
ajaxStop()	Specifies a function to run when all AJAX requests have completed
ajaxSuccess()	Specifies a function to run when an AJAX request completes successfully
load()	Loads data from a server and puts the returned data into the selected element

jQuery ajax() Method

- The ajax() method is used to perform an AJAX (asynchronous HTTP) request.
- All jQuery AJAX methods use the ajax() method. This method is mostly used for requests where the other methods cannot be used.
- Syntax:
  - $.ajax(*{name:value, name:value, ... }*)

jQuery ajax() Method

Name	Value/Description
url	Specifies the URL to send the request to. Default is the current page
data	Specifies data to be sent to the server
success(*result,status, xhr*)	A function to be run when the request succeeds
type	Specifies the type of request. (GET or POST)
dataType	The data type expected of the server response.

jQuery ajax() Method

- Example: Change the text of a element using an AJAX request:

```
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script>
    $(function () {
        $("button").click(function () {
            $.ajax({
                url: "../../demo_test.txt",
                success: function (result) {
                    $("#div1").append(result);
                }
            })
        });
    });
</script>
<div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>
<button>Get External Content</button>
```

được nhập:

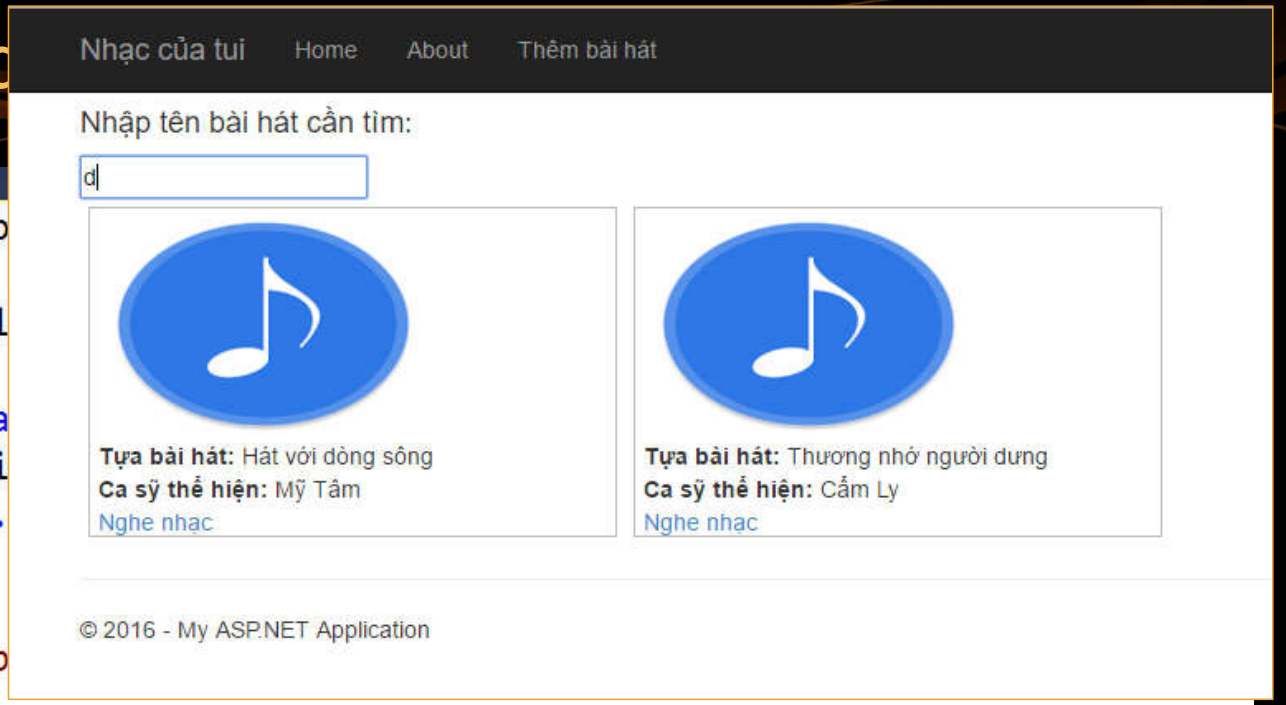
```
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script>
    $(function () {
        $("#search").keyup(function () {
            var search = $("#search").val();
            $.ajax({
                url: "/BaiHat/Search",
                data: { query: search },
                success: function (response) {
                    $("p#main").html(response);
                }
            });
        });
    });
</script>
<div>
    <h4>Nhập tên bài h
    <form>
        <input type="text" id="search" />
    </form>
    <p id="main">
    </p>
    <div style="clear:both"></div>
</div>
```

```
public ActionResult Search(string query)
{
    db=new QuanLyBaiHatEntities();
    List<BaiHat> list = db.BaiHats.Where(x =>
        x.tenBH.ToLower().Contains(query.ToLower())).ToList<BaiHat>();
    return PartialView(list);
}
```

jQuery ajax() Method

- Search.cshtml*

```
@model List<WebApplication1.Mo
@{
    foreach (var item in Model
    {
        <div id="productDispla
            <a href="@Url.Acti
                <img src="../.
            </a>
            <br />
            <b>Tựa bài hát:</b
            <br />
            <b>Ca sỹ thể hiện:</b> @item.CaSi.tenCS
            <br />
            <a href="@Url.Action("Details", "BaiHat", new { @id = item.maBH.Trim() })">
            Nghe nhạc</a>
        </div>
    }
}
```



jQuery ajaxSetup() Method

- The ajaxSetup() method sets default values for future AJAX requests.
- Syntax

```
$.ajaxSetup({name:value, name:value, ... })
```

jQuery ajaxSetup() Method

Name	Value/Description
dataType	The data type expected of the server response.
error(*xhr,status,error*)	A function to run if the request fails.
type	Specifies the type of request. (GET or POST)
url	Specifies the URL to send the request to. Default is the current page
data	Specifies data to be sent to the server
success(*result,status,xhr*)	A function to be run when the request succeeds

jQuery ajaxSetup() Method

- Example: Change the text of a element using an AJAX request:

```
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            $.ajaxSetup({
                url: "../../demo_ajax_load.txt", success: function (result) {
                    $("div").html(result);
                },
                error: function (xhr, status) { alert(status); }
            });
            $.ajax();
        });
    });
</script>
<div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>
<button>Get External Content</button>
```

jQuery get() Method

- The $.get() method loads data from the server using a HTTP GET request.
- Syntax
- $.get(*URL,data,function(data,status,xhr),dataType)*

jQuery get() Method

Parameter	Description
URL	Required. Specifies the URL you wish to request
data	Optional. Specifies data to send to the server along with the request
function(data,status,xhr)	•Optional. Specifies a function to run if the request succeeds Additional parameters: *data* - contains the resulting data from the request •*status* - contains the status of the request ("success", "notmodified", "error", "timeout", or "parsererror") •*xhr* - contains the XMLHttpRequest object
dataType	•Optional. Specifies the data type expected of the server response. By default jQuery performs an automatic guess. Possible types:"xml" - An XML document •"html" - HTML as plain text •"text" - A plain text string •"script" - Runs the response as JavaScript, and returns it as plain text •"json" - Runs the response as JSON, and returns a JavaScript object •"jsonp" - Loads in a JSON block using JSONP. Will add an "?callback=?" to the URL to specify the callback

```
public class HomeController : Controller
{
    public string GetToDay()
    {
        return DateTime.Today.ToString();
    }
    public ActionResult GetForm()
    {
        return View();
    }
```

ller's Method which will return

GetForm.cshtml*

```
    }
    <script src="~/Scripts/jquery-1.10.2.min.js"></script>
    <script>
        $(function () {
            $("#btnToDay").click(function () {
                $.get(
                    "/Home/GetToDay", null, function (data) {
                        $("#test").html(data);
                    });
            });
        });
    </script>
    <h2>GetForm</h2>
    <form >
        <input type="button" id="btnToDay" value="Get Today" />
    </form>
    <div id="test">
    </div>
```

r to Controller's Method

```
public class HomeController : Controller
{
    public string WelcomeMsg(string input)
    {
        if (!String.IsNullOrEmpty(input))
            return "Please welcome " + input;
        else
            return "Please ent
    }
    public ActionResult GetWit
    {
        return View();
    }
```

GetWithPara.cshtml*

```
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script>
    $(function () {
        $("#btnSubmit").click(function () {
            var name=$("#txtName").val();
            $.get("/Home/WelcomeMsg", { input: name },
                function (data) {
                    alert(data);
                    $("#data").html(data);
                });
        });
    });
</script><h2>GetWithPara</h2>
<form>
    @Html.Label("Your name: ") @Html.TextBox("txtName") <br />
    <input type="button" id="btnSubmit" value="Submit" />
</form>
<p id="data"></p>
```

jQuery post() Method

- The $.post() method loads data from the server using a HTTP POST request.
- Syntax
  - $(*selector*).post(*URL,data,function(data,status,xhr),dataType)*

jQuery post() Method

Parameter	Description
URL	Required. Specifies the url to send the request to
data	Optional. Specifies data to send to the server along with the request
function(data,status,xhr)	•Optional. Specifies a function to run if the request succeeds Additional parameters: *data* - contains the resulting data from the request •*status* - contains the status of the request ("success", "notmodified", "error", "timeout", or "parsererror") •*xhr* - contains the XMLHttpRequest object
dataType	•Optional. Specifies the data type expected of the server response. By default jQuery performs an automatic guess. Possible types:"xml" - An XML document •"html" - HTML as plain text •"text" - A plain text string •"script" - Runs the response as JavaScript, and returns it as plain text •"json" - Runs the response as JSON, and returns a JavaScript object •"jsonp" - Loads in a JSON block using JSONP. Will add an "?callback=?" to the URL to specify the callback

jQuery post() Method

- Example: **POST call to Con**
  (not form)

```
[HttpPost]
public string SubmitSubscription(string Name, string
{
    if (!String.IsNullOrEmpty(Name) && !String.IsNul
        //TODO: Save the data in database
        return "Thank you " + Name + ". Record Saved
    else
        return "Please complete the form.";

}
```

```
<h2>Subscription</h2>

<p>
    Enter your name
    <br />
    @Html.TextBox("Name")
</p>
<p>
    Enter your address
    <br />
    @Html.TextBox("Address")
</p>

<input type="button" value="Save" id="Save" />
<span id="msg" style="color:red;"/>

<script type="text/javascript">
    $('#Save').click(function () {
        var url = "/Home/SubmitSubscription";
        var name = $("#Name").val();
        var address = $("#Address").val();
        $.post(url, { Name: name, Address: address }, function (data) {
            $("#msg").html(data);
        });
    })
</script>
```

jQuery post() Method

- Example: **POST call to Contro**

```
public class Subscription
{
    public string Name { get; set; }
    public string Address { get; set; }
}
```

```
[HttpPost]
public string SubmitSubscription(Subscription subs)
{
    if (!String.IsNullOrEmpty(subs.Name) && !String.IsNullOrEm
        //TODO: Save the data in database
        return "Thank you " + subs.Name + ". Record Saved.";
    else
        return "Please complete the form.";

}
```

```
<h2>Subscription</h2>

<form id="subscriptionForm" action="/Home/SubmitSubscription" method="post">
<p>
    Enter your name
    <br />
    @Html.TextBox("Name")
</p>
<p>
    Enter your address
    <br />
    @Html.TextBox("Address")
</p>

<input type="button" value="Save" id="Save" />
<span id="msg" style="color:red;"/>
</form>

@section Scripts{
    <script type="text/javascript">
        $('#Save').click(function () {

            var form = $("#subscriptionForm");
            var url = form.attr("action");
            var formData = form.serialize();
            $.post(url, formData, function (data) {
                $("#msg").html(data);
            });
        })
    </script>
}
```

jQuery getScript() Method

- The getScript() method is used to get and execute a JavaScript using an AJAX HTTP GET request.
- Syntax
  - $(*selector*).getScript(*url,success(response,status))*

jQuery load() Method

- The load() method loads data from a server and puts the returned data into the selected element.
- **Note:** There is also a jQuery Event method called load. Which one is called, depends on the parameters.
- Syntax
  - $(*selector*).load(*url,data,function(response,status,xhr*))

jQuery load() Method

Parameter	Description
url	Required. Specifies the URL you wish to load
data	Optional. Specifies data to send to the server along with the request
function(response,status,xhr)	•Optional. Specifies a callback function to run when the load() method is completed. Additional parameters: *response* - contains the result data from the request •*status* - contains the status of the request ("success", "notmodified", "error", "timeout", or "parsererror") •*xhr* - contains the XMLHttpRequest object

jQuery load() Method

```
<h2>LoadMethod</h2>
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("#div1").load("../../TextFile1.txt");
        });
    });
</script>
<div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>

<button>Get External Content</button>
```

JSON Services in ASP.NET MVC

JSON Services in ASP.NET MVC

- MVC Ajax Helpers cover simple AJAX scenarios
  - Replacing HTML content
  - Partial page rendering
- Other scenarios require some JavaScript
  - Auto-complete textboxes
  - Client-side validation
  - Invoking JSON services and render content at the client-side
  - Animations

jQuery getJSON() Method

- The getJSON() method is used to get JSON data using an AJAX HTTP GET request.
- Syntax
- $(*selector*).getJSON(*url,data,success(data,status,xhr)*)

jQuery getJSON() Method

Parameter	Description
url	Required. Specifies the url to send the request to
data	Optional. Specifies data to be sent to the server
success(data,status,xhr)	•Optional. Specifies the function to run if the request succeeds Additional parameters:*data* - contains the data returned from the server. •*status* - contains a string containing request status ("success", "notmodified", "error", "timeout", or "parsererror"). •*xhr* - contains the XMLHttpRequest object

MVC, JSON Results and JavaScript Rendering

- Return `JsonResult` in the action

```
public JsonResult AllBooks()
{
    var books = BooksData.GetAll();
    return this.Json(books, JsonRequestBehavior.AllowGet);
}
```

- Use `jQuery.getJSON(…)`

```
jQuery.getJSON("AllBooks", null, function(data) {
    jQuery(data).each(function (index, element) {
        var newBookElement = $("<li>+element.Title+"</li>");
        $("#books").append(newBookElement);
    });
});
```

jQuery getJSON() Method

- Exar

```
public class HomeController : Controller
{
    QuanLyBaiHatEntities db = new QuanLyBaiHatEntities();
    public ActionResult GetBaiHatTheoCaSi()
    {
        List<CaSi> lst = db.CaSis.ToList<CaSi>();
        ViewBag.maCS = new SelectList(lst, "maCS", "tenCS");
        return View();
    }
    public ActionResult DanhSachBaiHatTheoCaSi(String macs)
    {
        db = new QuanLyBaiHatEntities();
        var kq = db.BaiHats.Where(x => x.maCS.Trim() == macs).
            Select(x=>new {maBH=x.maBH,tenBH=x.tenBH});
        return Json(kq, JsonRequestBehavior.AllowGet);
    }
```

jQuery getJSON() Method

```
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script>
    $(function () {
        $("#maCS").change(function () {
            var Id = $("#maCS").val().trim();
            $.getJSON("/Home/DanhSachBaiHatTheoCaSi", "macs=" + Id, function (data) {
                $("#ds").html("");
                $.each(data, function (i, bh) {
                    $("#ds").append("<option value='" + bh.maBH + "'> " + bh.tenBH + "</option>"
                })
            });
        });
    });
</script>
```

```
<form >
    <h2>Hãy chọn ca sĩ:</h2>
    @Html.DropDownList("maCS", "Chọn ca sĩ")
    <p></p>
    <select id="ds"></select>
</form>
```

AJAX with Unobtrusive JavaScript

AJAX with Unobtrusive JavaScript & jQuery

- Unobtrusive JavaScript == no script is injected into page
  - Only data-attributes to configure the AJAX call settings
- Requires Microsoft.jQuery.Unobtrusive.Ajax NuGet package
  - **`jquery.unobtrusive-ajax.js`** (AJAX helpers)

```
<a data-ajax="true" href="/Home/ServerTime"
  data-ajax-method="GET" data-ajax-mode="replace"
  data-ajax-update="#timeDisplay">
    Load Server Time
</a>
```

AJAX Helpers in ASP.NET MVC

@Ajax.ActionLink and @Ajax.BeginForm

AJAX Helpers in ASP.NET MVC

- AJAX helpers add AJAX functionality to ASP.NET MVC
- Two core features of AJAX helpers:
  - Invoke an action method asynchronously using AJAX
    - Use the `Ajax.ActionLink()` helper
  - Submit an entire form using AJAX
    - Use the `Ajax.BeginForm()` helper
- AJAX helpers use `AjaxOptions` object with configurations

AjaxOptions Object

- **`Url`** – URL to send request
- **`HttpMethod`** – request method (GET / POST / PUT / DELETE / …)
- **`InsertionMode`** – how to handle the received data
  - **`InsertAfter`**, **`InsertBefore`** or **`Replace`**
- **`UpdateTargetId`** – HTML element to be changed
- **`LoadingElementId`** – show / hide "Loading…" when loading
- Events (JavaScript functions)
  - **`OnSuccess`**, **`OnFailure`**, **`OnBegin`**, **`OnComplete`**

Ajax.ActionLink Helper – Example

- Defines an action link (`<a href=…>`) for loading data with AJAX

```
@Ajax.ActionLink("Load Server Time", "ServerTime", null,
  new AjaxOptions {
    HttpMethod = "GET",
    UpdateTargetId = "timeDisplay",
    LoadingElementId = "timeDisplayLoading",
    InsertionMode = InsertionMode.Replace,
    OnBegin = "OnAjaxRequestBegin",
    OnFailure = "OnAjaxRequestFailure",
    OnSuccess = "OnAjaxRequestSuccess",
    OnComplete = "OnAjaxRequestComplete",
}, new { @class = "btn btn-primary" })
```

action

controller

Ajax.BeginForm Helper – Example

- A form that submits AJAX request and renders a partial view

```
@using (Ajax.BeginForm("Search",
    new AjaxOptions {
        UpdateTargetId = "results",
        InsertionMode = InsertionMode.Replace
    }))
{
    <input type="text" name="query" />
    <input type="submit" />
}

<div id="results">@Html.Partial("_BookResult", Model)</div>
```

Ajax.BeginForm Helper – Example (2)

- Return a **PartialView** to the helpers (view without the layout)

```
public ActionResult Search(string query)
{
  var result = BooksData
    .GetAll()
    .AsQueryable()
    .Where(book => book.Title.Contains(query))
    .Select(BookViewModel.FromBook)
    .ToList();

  return this.PartialView("_BookResult", result);
}
```

Summary

- AJAX is powerful technique in Web development
  - Load data asynchronously
  - Without refreshing the entire Web page
- AJAX Helpers in ASP.NET MVC simplify AJAX calls
  - Controllers return partial views, displayed on the Web page
  - Use `ActionLink` and `BeginForm` for simple AJAX calls
- Still we can implement JSON service with client-side rendering
  - Typically use jQuery AJAX functionality

Ajax

AJAX in ASP.NET MVC