CODER FOUNDRY

Using jQuery DataTables

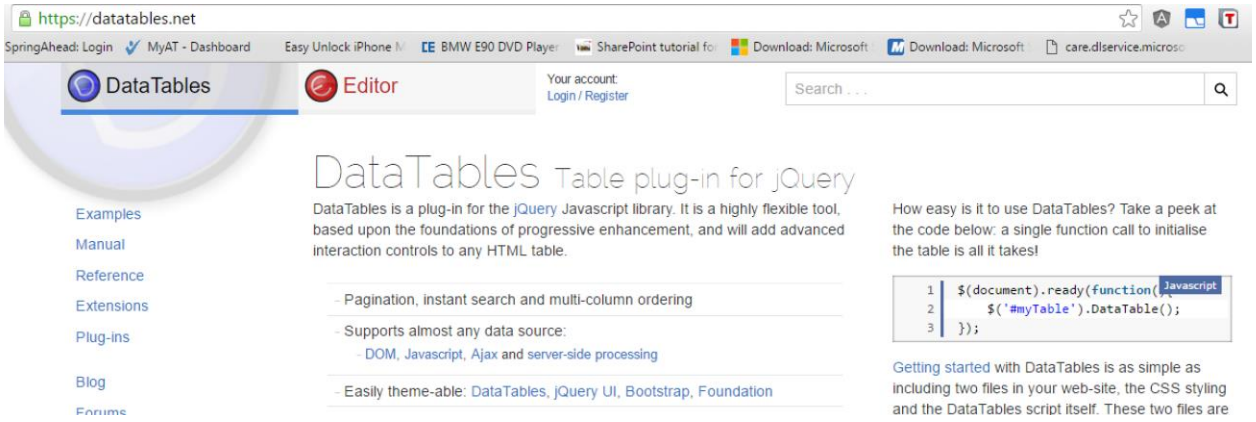
HTML tables, as rendered through scaffolding in MVC, are not the prettiest things in the world, and they lack a certain degree of user interactivity. Ok, let's face it, there is no user interactivity.

What if we want to be able to quickly and easily sort (ascending and descending) by column headings, include full-text search on the table contents, add paging to the table, and let the user decide how many records to display on a given page? To add this functionality using MVC and Razor syntax is an exercise in near-futility that results in a complex interweaving of controller actions, helper methods, myriad optional parameters, and a thoroughly annoying full page refresh after every user selection.

Wouldn't it be great if there was a way to do all of that without having to jump through hoops a circus performer would be afraid of?

Enter jQuery DataTables!

https://datatables.net/download/index



DataTables Table plug-in for jQuery

DataTables is a plug-in for the jQuery Javascript library. It is a highly flexible tool, based upon the foundations of progressive enhancement, and will add advanced interaction controls to any HTML table.

- Pagination, instant search and multi-column ordering
- Supports almost any data source:
  - DOM, Javascript, Ajax and server-side processing
- Easily theme-able: DataTables, jQuery UI, Bootstrap, Foundation

How easy is it to use DataTables? Take a peek at the code below: a single function call to initialise the table is all it takes!

```
$(document).ready(function(){
    $('#myTable').DataTable();
});
```

Getting started with DataTables is as simple as including two files in your web-site, the CSS styling and the DataTables script itself. These two files are

There are many available JavaScript plug-ins that do this sort of thing, but jQuery DataTables offers the most bang for your buck (well, not really, it's free). It works with your existing scaffolded MVC table structure, only requires a few small additions to your code, and requires no translation of your model data into JSON strings, XML, or other similar annoyances.

With that in mind, let's show you how to set up your project to work with jQuery DataTables.

1) To include DataTables on your page simply include the following HTML:

```
<!-- CSS -->
jquery.dataTables.min.css


<!-- jQuery -->
jquery.min.js


<!-- DataTables-->
jquery.dataTables.min.js
```

2) Decide where to load the resources (_Layout or specific View)

As with any JavaScript library dependent on jQuery, it is imperative that jQuery be loaded first. You need to decide whether to load the DataTables resources for every page in your project – which means you must load it in the _Layout.cshtml page – or for individual views. (You could also set up a second layout based on the first that includes the DataTables references, then use that layout only for those pages that need DataTables.)

Once you load the DataTables resources, be sure that you load jQuery first!!! Order matters!!

3) Modify your table

DataTables require the addition of two tags, <thead> and <tbody>, to the existing MVC-rendered tables in your View. This is an example:

```
<table id="mytable" class="table">

    <thead>
        <tr>
            <th>
                 ...
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model) {
            <tr>
                <td>
                    ...
                </td>
                <td>
                    @Html.ActionLink("Edit", "Edit", new { id=item.Id }) |
                    @Html.ActionLink("Details", "Details", new { id=item.Id }) |
                    @Html.ActionLink("Delete", "Delete", new { id=item.Id })
                </td>
            </tr>
        }
    </tbody>
</table>
```

4) Now we need to Apply the DataTable to your existing table

Next, to enable datatables (in similar fashion of CKeditor & TinyMCE) use the table’s id attribute, as shown here:

```
<script>
    $('#mytable').DataTable();
</script>
```

You can also reference your table(s) by class or by tag type. This is useful if you have more than one table on a page that you need to render:

```
<script>
    // format ALL tables by class selector
    $('.table').DataTable();
</script>
```

5) That’s it! You can now test your newly formatted interactive tables on your web page.