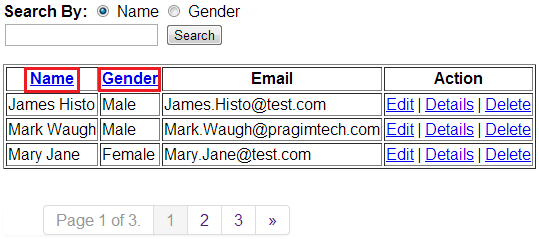
In this video, we will discuss, **implementing sort functionality in an asp.net mvc application**. We will be working with the example that we started in [Part 62](http://csharp-video-tutorials.blogspot.com/2013/07/part-62-implementing-search.html). So, please watch [Parts 62](http://csharp-video-tutorials.blogspot.com/2013/07/part-62-implementing-search.html) and [63](http://csharp-video-tutorials.blogspot.com/2013/07/part-63-implement-paging-in-aspnet-mvc.html) before proceeding with this video.

**We want to support bi-directional sorting by Name and Gender columns. Here's the requirement**  
**1.** **Name & Gender** columns must be **click-able hyperlinks**  
**2.** Clicking on the column headers should sort the data. If the data is not already sorted by the column on which you have clicked, the data should be sorted in ascending order. Clicking again on the same column should sort the data in descending order.  
**3.** By default, the data should be sorted by **"Name"** in ascending order.

By the end of this video, the output should be as shown below. Notice that **"Name"** and **"Gender"** columns are rendered as hyperlinks, which the user can click to sort data.   
   
  
**Step 1:** Modify the **"Index()"** action method in **HomeController** as shown below.  
public ActionResult Index(string searchBy, string search, int? page, string sortBy)  
{  
    ViewBag.NameSort = String.IsNullOrEmpty(sortBy) ? "Name desc" : "";  
    ViewBag.GenderSort = sortBy == "Gender" ? "Gender desc" : "Gender";  
  
    var employees = db.Employees.AsQueryable();  
              
    if (searchBy == "Gender")  
    {  
        employees = employees.Where(x => x.Gender == search || search == null);  
    }  
    else  
    {  
        employees = employees.Where(x => x.Name.StartsWith(search) || search == null);  
    }  
  
    switch (sortBy)  
    {  
        case "Name desc":  
            employees = employees.OrderByDescending(x => x.Name);  
                break;  
        case "Gender desc":  
                employees = employees.OrderByDescending(x => x.Gender);  
                break;  
        case "Gender":  
                employees = employees.OrderBy(x => x.Gender);  
                break;  
        default:  
            employees = employees.OrderBy(x => x.Name);  
                break;  
    }  
  
    return View(employees.ToPagedList(page ?? 1, 3));  
}  
  
**Step 2:** Modify the code in **Index.cshtml** view as shown below. Please pay attention to the code highlighted with **Grey** colour.  
@using PagedList;  
@using PagedList.Mvc;  
  
@model PagedList.IPagedList<MVCDemo.Models.Employee>  
  
@{  
    ViewBag.Title = "Index";  
}  
<link href="~/Content/PagedList.css" rel="stylesheet" type="text/css" />  
<div style="font-family:Arial">  
<h2>Employee List</h2>  
<p>  
    @using (@Html.BeginForm("Index", "Home", FormMethod.Get))  
    {  
        <b>Search By:</b>  
        @Html.RadioButton("searchBy", "Name", true) <text>Name</text>  
        @Html.RadioButton("searchBy", "Gender") <text>Gender</text><br />

 @Html.TextBox("search") <input type="submit" value="search" />  
          
    }  
</p>  
<table border="1">  
    <tr>  
        <th>  
            @Html.ActionLink("Name", "Index", new { sortBy = ViewBag.NameSort, searchBy = Request["searchBy"], search = Request["search"] })  
        </th>  
        <th>  
            @Html.ActionLink("Gender", "Index", new { sortBy = ViewBag.GenderSort, searchBy = Request["searchBy"], search = Request["search"] })  
        </th>  
        <th>  
            @Html.DisplayNameFor(model => model.First().Email)  
        </th>  
        <th>Action</th>  
    </tr>  
@if (Model.Count() == 0)  
{  
    <tr>  
        <td colspan="4">  
            No records match search criteria  
        </td>  
    </tr>  
}  
else  
{  
    foreach (var item in Model)  
    {  
        <tr>  
            <td>  
                @Html.DisplayFor(modelItem => item.Name)  
            </td>  
            <td>  
                @Html.DisplayFor(modelItem => item.Gender)  
            </td>  
            <td>  
                @Html.DisplayFor(modelItem => item.Email)  
            </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>  
    }  
}  
</table>  
@Html.PagedListPager(Model, page => Url.Action("Index", new { page, searchBy = Request.QueryString["searchBy"], search = Request.QueryString["search"], sortBy = Request["sortBy"] }), new PagedListRenderOptions() { Display = PagedListDisplayMode.IfNeeded })  
</div>